

**UNCLASSIFIED**

---

**AD 295 501**

---

*Reproduced  
by the*

**ARMED SERVICES TECHNICAL INFORMATION AGENCY  
ARLINGTON HALL STATION  
ARLINGTON 12, VIRGINIA**



---

**UNCLASSIFIED**

**Best  
Available  
Copy**

NOTICE: When government or other drawings, specifications or other data are used for any purpose other than in connection with a definitely related government procurement operation, the U. S. Government thereby incurs no responsibility, nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use or sell any patented invention that may in any way be related thereto.

63-2-3

CATALOGUE OF ASTIA  
AS AD NO.

5

295 501

**SDC**

TM-555/020 00

JOVIAL Generator Description:

1 July 196



SYSTEM DEVELOPMENT CORPORATION  
2500 COLORADO AVENUE  
SANTA MONICA, CALIFORNIA

TRANS. NO. U 28737

## DOCUMENT TRANSMITTAL

DATE	1-22-63	PAGE	1	OF	1	PAGES	<input type="checkbox"/> AIRMAIL	<input type="checkbox"/> FIRST CLASS
FROM		TO						
SYSTEM DEVELOPMENT CORPORATION 2500 COLORADO AVENUE SANTA MONICA, CALIFORNIA		ASPCA Arlington Hall Station Arlington 12, Virginia						
IDENTIFICATION	NO. OF COPIES	TITLE AND/OR SUBJECT (if necessary)						
UNCLASS.	10	SM 555/000/00  The enclosed documents have been cleared for open publication. This material may be issued without restrictions or limitations and may be listed in the white section of TMS.						
THE MATERIAL LISTED ABOVE IS BEING TRANSMITTED FOR YOUR USE AND RETENTION IN RESPONSE TO:								
<input type="checkbox"/> INITIAL DISTRIBUTION LIST								
<input type="checkbox"/> YOUR REQUEST _____								
							SYSTEM DEVELOPMENT CORPORATION <i>E. A. Green / H.</i> Signature	

# TECHNICAL MEMORANDUM

(TM Series)

---

JOVIAL Generator Description

by

Virginia L. Cohen  
Millard H. Perstein

10 July 1962

SYSTEM

DEVELOPMENT

CORPORATION

2500 COLORADO AVE.

SANTA MONICA

CALIFORNIA

---

The views, conclusions, or recommendations expressed in this document do not necessarily reflect the official views or policies of agencies of the United States Government.

Permission to quote from this document or to reproduce it, wholly or in part, should be obtained in advance from the System Development Corporation.



10 July 1962

1  
(page 2 blank)

TM-555/020/00

## TABLE OF CONTENTS

	<u>Page</u>
INTRODUCTION . . . . .	3
DESCRIPTION OF PHASE I OF THE JOVIAL GENERATOR	
A. General Description . . . . .	7
B. Glossary (Item, Table, Procedure, Switch, and File Descriptions) .	11
C. Descriptions of the GEN1 Main Program (MP) Regions. . . . .	65
D. Description of the GEN1 Procedures. . . . .	77
E. Error Message Description . . . . .	113
DESCRIPTION OF PHASE II OF THE JOVIAL GENERATOR	
A. General Description . . . . .	119
B. Glossary (Item, Table, Procedure, Switch, and File Descriptions) .	125
C. Descriptions of the GEN2 Main Program (MP) Regions. . . . .	181
D. Description of the GEN2 Procedures. . . . .	201
E. Error Message Description . . . . .	289

AD 295501

8 August 1962

2A  
(Page 2B blank)

TM-555/020/00A

## MODIFICATION TO:

TM-555/020/00, "JOVIAL Generator

Description" dated 10 July 1962



APPROVED

*M. H. Perstein*

M. H. Perstein

System Development Corporation / 2880 Colorado Ave. / Santa Monica, California

### CURRENT MODIFICATIONS

#### Modified Pages

#### Notes and Filing Instructions

2A  
141  
149  
182  
184  
191  
193  
208  
238  
290  
293

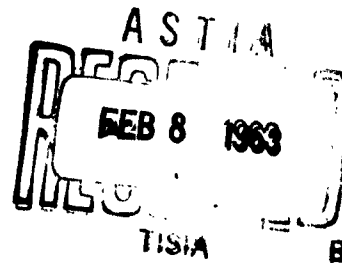
Additions and modifications to Phase II of the JOVIAL Generator Description. Please remove pages 141, 184, 191, 208, 238 and 293 dated 10 July 1962. Insert pages 2A, 141, 184, 191, 208, 238, 293 and 294 dated 8 August 1962.

ERRATA\* Page 149 dated 10 July 1962. Change last two digits of line 3 of ILY so it reads as follows: "Meaningful only if ILC EQ 0 and ILO EQ 25."

ERRATA\* Page 182 dated 10 July 1962. Change line 1 of region two to read as follows: " 2. a. Name - NEXT1"

ERRATA\* Page 290 dated 10 July 1962. Strike out words "Detected by: Procedure DESIG" from error number 110.

ERRATA\* Page 193 dated 10 July 1962. In section 20.d.1.2.5.3, change line 2 to read "continue at d.1.2.5"





18 September 1962

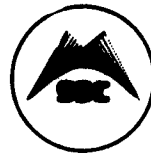
2A  
(Page 2B blank)

TM-555/020/00B

# MODIFICATION TO:

TM-555/020/00. "JOVIAL Generator

Description" dated 10 July 1962



APPROVED

N. J. Cohen

V.L. Cohen

M.H. Perstein

System Development Corporation/2800 Colorado Ave./Santa Monica, California

## CURRENT MODIFICATIONS

### Modified Pages

### Notes and Filing Instructions

2A  
13  
18  
29  
41  
98  
115  
118  
132  
149  
235  
291

Additions and modifications to JOVIAL Generator Description. Please remove pages 13, 18, 29, 41, 98, 115, 118, 132, 149 and 235 dated 10 July 1962.

ERRATA\* Page 291 dated 10 July 1962. Delete words "(not used)" from error number 123 and add the following: "NENT, ENTRY, or NWDSEN modifier missing".

## CUMULATIVE LIST OF MODIFICATIONS

### Modified Pages

### Modification Number and Date

### Modified Pages

### Modification Number and Date

2A	B (9/18/62)	New Page
13	B "	" "
18	B "	" "
29	B "	" "
41	B "	" "
98	B "	" "
115	B "	" "
118	B "	" "
132	B "	" "
141	A (8/8/62)	" "
149	B (9/18/62)	" "

182	A (8/8/62)	Errata
184	A "	New Page
191	A "	" "
193	A "	Errata
208	A "	New Page
235	B (9/18/62)	" "
238	A (8/8/62)	" "
290	A "	Errata
291	B (9/18/62)	Errata
293	A (8/8/62)	New Page

\*ERRATA modifications are to be entered by hand.

10/22/62

16 October 1962

2A  
(Page 2B Blank)

TM-555/020/00C

# MODIFICATION TO:

TM-555/020/00. "JOVIAL Generator

Description" dated 10 July 1962



APPROVED

*V. L. Cohen* *M. H. Perstein*  
V. L. Cohen M. H. Perstein

System Development Corporation/2880 Colorado Ave./Santa Monica, California

## CURRENT MODIFICATIONS

### Modified Pages

### Notes and Filing Instructions

30 Additions and modifications to JOVIAL Generator Description  
31 Please remove pages 30, 31 and 32 dated 10 July 1962.  
32 Insert pages 2A, 30, 31 and 32 dated 16 October 1962.  
149  
150  
235  
291  
ERRATA\* Page 149 dated 18 September 1962. Change the last two digits of line 3 of ILY so it reads as follows: "Meaningful only if ILC EQ 0 and ILO EQ 25."  
ERRATA\* Page 150 dated 10 July 1962. Change the word tape to type in line 2, it should read as follows: Sensitive to the type (AA) of processing to be performed by the procedure.  
ERRATA\* Page 235 dated 18 September 1962. In section 4.h.1. change line 1 to read: If the first entry examined is a comma or a right parenthesis  
ERRATA\* Page 291 dated 10 July 1962. In line 123: Insert ALL, in front of NENT.

## CUMULATIVE LIST OF MODIFICATIONS

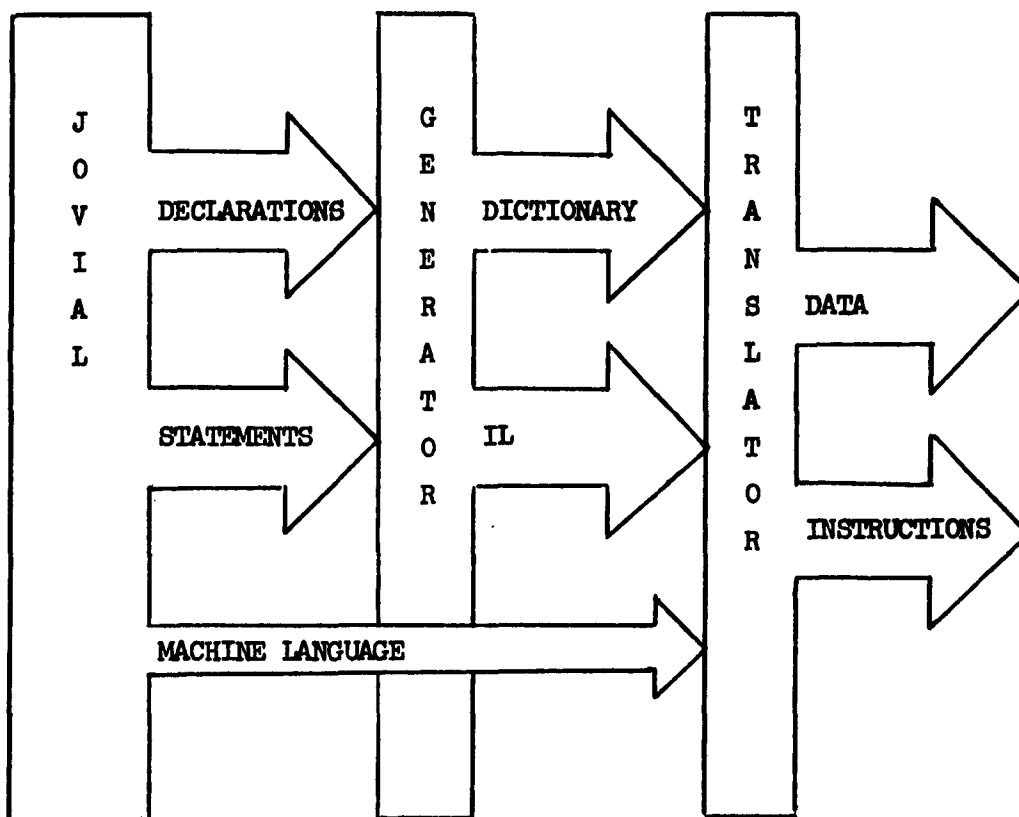
Modified Pages	Modification Number and Date	Modified Page	Modification Number and Date
2A	C (10/16/62) New Page	149	C (10/16/62) Errata
13	B (9/18/62) " "	150	C " "
18	B " " "	182	A (8/8/62) "
29	B " " "	184	A " New Page
30	C (10/16/62) " "	191	A " " "
31	C " " "	193	A " Errata
32	C " " "	208	A " New Page
41	B (9/18/62) " "	235	C (10/16/62) Errata
98	B " " "	238	A (8/8/62) New Page
115	B " " "	290	A " Errata
118	B " " "	291	C (10/16/62) "
132	B " " "	293	A (8/8/62) New Page
141	A (8/8/62) " "		
149	B (9/18/62) " "		

\*Errata modifications are to be entered by hand.

## INTRODUCTION

A compiler is a program which translates a higher level language called the source language, in which the solution to a problem is expressed, into a machine code representation of this solution called the object or target language.

It is readily apparent that a compiler is an automatic coding device which performs a necessary, albeit tedious, part of the task of obtaining a machine solution. Since compilers are expensive programs to write and JOVIAL compilers were to exist on several machines, they were divided into two parts. The first part is called the generator, and is essentially the same program for all computers. The second part is called the translator and a new one is written for each computer. The generator inputs JOVIAL source language and produces the intermediate language and dictionaries. The translator takes the output of the generator and outputs machine code. The generator is actually divided into two passes which are described in detail on the following pages.

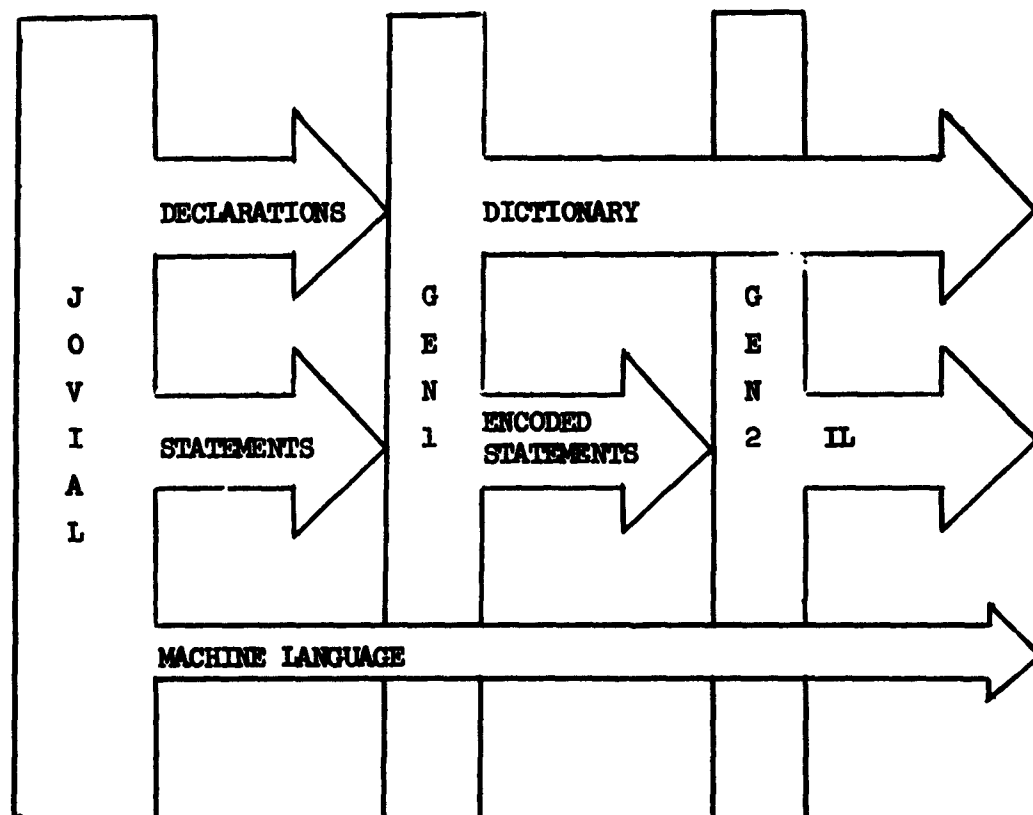
OVERVIEW OF THE JOVIAL-3 COMPILERS

10 July 1962

5  
(page 6 blank)

TM-555/020/00

OVERVIEW OF THE GENERATOR





## DESCRIPTION OF PHASE I OF THE JOVIAL GENERATOR

## A. GENERAL DESCRIPTION

Generator Phase I looks at the JOVIAL source program and encodes numerically the information contained in the source program. This information is stored in a series of tables which are passed on to Generator Phase II for further processing. Phase I is made up of a series of procedures which are described below, and a main program.

Generator Phase I may be divided into two sections; the first processes operators, separators, and brackets; i.e., those parts of speech which are immediately identifiable. The second section processes identifiers (item names, table names, etc.) and constants.

The JOVIAL source language program is processed one part of speech at a time, either by section one or two, depending upon the kind of part of speech encountered. As each part of speech is processed, an entry is made for it in the Card Analysis Table (CAT). If the part of speech is an identifier processed by the second section of Phase I, a reference to an entry in the Dictionary Table (DICT) is put in CAT, and in some cases entries are made in other tables described below. As each new identifier is encountered by section two, an entry is made in DICT.

The final output of Phase I is the CAT, DICT and other related tables. The CAT contains a binary encoding of each part of speech in the source program. In other words, CAT may be thought of as a binary replica of the statements in the source program. DICT contains information about the identifiers and constants used in the source program (type of constant, number of bits in a variable, etc.).

Each part of speech being processed is viewed as the middle term of a set of three; the left term of the triplet having been processed already and the right term waiting to be processed next. When a declaration is encountered, the triplet scheme is abandoned temporarily and the declaration is processed until its end when the triplet method is resumed. For each possible left and right term combination there is an entry in the triplet legality matrix table (MTRX or MAT). This entry indicates what kind of term may be found legally as the middle term of the triplet in question. For example, the first section of Phase I can detect a procedure name using the triplet scheme. The information that a procedure name has been found is passed on to section two, and the appropriate reference in the dictionary is found and returned to be stored in CAT. However, section one cannot discriminate between a simple item, a file name, the beginning of a declaration etc. As much information as is available is passed on to section two and further analysis is done, and the resulting dictionary reference returned.

The entire JOVIAL source program is processed in this manner, with special consideration being given to DIRECT code statements, DEFINE declarations, etc.

#### INPUT/OUTPUT

The input to Generator Phase I is the JOVIAL source language program which is to be compiled, read in from the HIN.

The output from Generator Phase I is the tables CAT, DICT, OVER, DISH and the STC variable IDNS, the direct code file DIRCT, and a listing of the JOVIAL source program which includes error printouts. The listing may also contain a print of CAT and/or DICT if indicated.

#### RESTRICTIONS

1. The following tables are restricted to the number of entries specified under Machine Dependencies:

	PURE (number of library procedures)
unnamed table with items KDA, KDB, KDC	(number of define declarations)
	FIND (level of nesting of modifiers)
	STAT (number of status constants)
	OVER (number of overlay declarations)
	DICT (number of dictionary entries)
unnamed table with item	CARD (number of characters of defined information which may be nested)

2. The following variables are restricted to the number of characters specified under Machine Dependencies:

IDNS	(all identifiers and constants)
KD	(contains defined identifiers and their definitions)

3. A Hollerith or STC variable with preset data is restricted to 120 characters.
4. All procedure declarations must be bracketed with BEGIN - END; i.e., even a one statement procedure must have brackets.
5. No circular define statements are permissible; i.e., the following is not allowable:

```

DEFINE AA  'CC = DD $ ' ' $
DEFINE DD  'AA + BB ' ' $

      or

DEFINE XX  'XX ' ' $

```



10 July 1962

9  
(page 10 blank)

TM-555/020/00

6. The ASSIGN and JOVIAL operators must be preceded by a blank column, or begin in the first column of a card.

10 July 1962

11

~~TM~~-555/020/00

## B. GLOSSARY

Item, Table, Procedure, Parameter, Switch, and File Descriptions

AA	Input Parameter	Local	Used by:	BTOD
	Input parameter which contains number to be converted.			
AA	Input Parameter	Local	Used by:	DTOB
	Input parameter which contains number to be converted.			
AA	Item	Local	Used by:	PDICT
	Contains the first dictionary entry to be printed.			
AA	Item	Local	Used by:	MUV
	Keeps track of entry in table overlayed with IDNT.			
AA	Item	Local	Used by:	TSTIM
	Keeps track of entry in table overlayed with TSTR.			
ABLE	Switch	Local	Used by:	LOOK
	Switch entered with input parameter to LOOK.			

0 = SWO (not used)  
1 = SWO (not used)  
2 = NORM (get next part of speech)

BCAT	File	Express	Used by: Main program
	File on which the binary CAT table for main program is output.		
BCATP	File	Express	Used by: Main program
	File on which the binary CAT table for procedures is output.		
BCE	Table	Contains items: DELIM, EXTRA, CLASS, FORM, KVAL, WOHIN	
		Express	Used by: Main program
	Table which contains class, form, KI value etc. for separators, operators, brackets, and modifiers etc.		
BF	Item	Local	Used by: CAN
	In processing, if decimal points are in the constant, they are removed. BF is used as a buffer for transferring a portion of the constant and overlaying the period while keeping track of its position for the E factor.		
BIND	Subscripted item from table FIND	Express	Used by: Main program
	Item in table FIND which is used to store counts of parenthesis while analyzing nested functional modifiers.		
BINRY	Output Parameter	Local	Used by: BINRY
	Variable containing the output of the function BINRY		
BINRY	Procedure		
	Procedure which converts an STC integer to binary.		
BKCT	Item	Express	Used by: ENTER, Main program
	Used to count number of parenthesis and subscript brackets when processing switch declarations.		
BLI	Item	Express	Used by: MP
	Boolean item used when processing direct code and scanning card image for ASSIGN or JOVIAL operators. Set to 0 when a blank is found and to 1 when a non-blank is found.		
BRCN	Subscripted item from unnamed table	Local	Used by: LOOK
	Holds bracket count for one type of bracket as each JOVIAL statement is processed.		
BRCTR	Item	Express	Used by: Main program
	Counter for BEGINS and ENDS in procedure.		

TM-555/020/00

### Procedure which converts a binary integer to STC.

**CAN** Procedure  
Procedure which analyzes constants and exits with the constant converted to conform with its TYPV or the TYPV of the item with which it is associated.

**CARD** Subscripted item from unnamed table Express Used by: LOOK, TSTIM, Main program  
Contains one character in STC from the card image read into TSTR.

**CAT** Table Contains items: CATA, CATB, CATC, CATE, CATF  
Express Used by: MP  
Table which contains binary encoding of the JOVIAL source program.

**CATA** Subscripted item from table CAT Express Used by: MP  
Boolean item which contains sign. Used primarily by Generator Phase II.  
0 = +  
1 = -

**CATB** Subscripted item from table CAT Express Used by: MP, RECRD  
Contains modifier.  
0 = NULL      4 = CHAR      8 = POS  
1 = NENT      5 = MANT      9 = SIGN  
2 = ENT      6 = ABS      10 = ODD  
3 = NWDSN      7 = ALL

**CATC** Subscripted item from table CAT Express Used ' : ENTER, LOOK, MP, RECRD, TSTIM  
Contains class. See CATF for details.

**CATE** Subscripted item from table CAT Express Used by: MP  
Contains level in Generator Phase II. Used only with class of 17 in Generator I when it contains number of cards of direct code.

**CAN** Procedure  
 Procedure which analyzes constants and exits with the constant converted to conform with its TYPV or the TYPV of the item with which it is associated.

**CARD** Subscripted item from unnamed table Express Used by: LOOK, TSTIM, Main program  
 Contains one character in STC from the card image read into TSTR.

**CAT** Table Contains items: CATA, CATB, CATC, CATE, CATF  
 Express Used by: MP  
 Table which contains binary encoding of the JOVIAL source program.

**CATA** Subscripted item from table CAT Express Used by: MP  
 Boolean item which contains sign. Used primarily by Generator Phase II.  
 0 = +  
 1 = -

**CATB** Subscripted item from table CAT Express Used by: MP, RECRD  
 Contains modifier.  
 0 = NULL      4 = CHAR      8 = POS  
 1 = NENT      5 = MANT      9 = SIGN  
 2 = ENT      6 = ABS      10 = ODD  
 3 = NWDSN      7 = ALL

**CATC** Subscripted item from table CAT Express Used by: ENTER, LOOK, MP, RECRD, TSTIM  
 Contains class. See CATF for details.

**CATE** Subscripted item from table CAT Express Used by: MP  
 Contains level in Generator Phase II. Used only with class of 17 in Generator I when it contains number of cards of direct code.

10 July 1962

16

TM-555/020/00

CATF

Subscripted item from table CAT

Express Used by: MP, ENTER,  
RECRD, TSTIM,  
LOOK

Contains form.

CLASS

0 = NULL  
1 = SLAB  
2 = FILE  
3 = CONS  
4 = VARB  
5 = SVAR  
6 = PROC  
7 = SUBS  
8 = SEQU

9 = SEPR

10 = BRAK

11 = LOGO

12 = RELO

13 = ARI0

14 = DECL

FORM

Channel in dictionary  
Channel in dictionary  
Channel in dictionary  
Channel in dictionary  
Channel in dictionary  
Channel in dictionary  
1 thru 26

0 = IF  
1 = IFEITH  
2 = ORIF  
3 = GOTO  
4 = FOR  
5 = RETURN  
6 = STOP  
7 = TEST

0 = .  
1 = ,  
2 = =  
3 = ==  
4 = \$  
5 = ...

0 = (  
1 = )  
2 = (\$  
3 = \$)  
4 = BEGIN  
5 = END  
6 = START  
7 = TERM  
8 = RESERVED  
9 = DOUBLE PRIME  
10 = RESERVED  
11 = RESERVED

0 = AND  
1 = OR  
2 = NOT

0 = LS  
1 = EQ  
2 = LQ  
3 = GR  
4 = NQ  
5 = GQ

0 = +  
1 = -  
2 = \*  
3 = /  
4 = \*\*  
5 = OF  
6 = MINUS  
7 = ABS

0 = SWITCH  
1 = PROC  
2 = CLOSE  
3 = ITEM  
4 = TABLE  
5 = ARRAY  
6 = OVERLAY  
7 = FILE  
8 = MODE  
9 = DEFINE  
10 = STRING  
11 = ALPHA



15 = IOUT

0 = INPUT            3 = OPEN OUTPUT  
 1 = OUTPUT          4 = SHUT INPUT  
 2 = OPEN OUTPUT    5 = SHUT OUTPUT

16 = BEAD

0 = BYTE  
 1 = BIT  
 0 = BEAD

17 = DIRECT

FIRST CARD NUMBER (CATE = NUMBER OF CARDS)

18 = TEMP

19 = GLAB

20 = A(

BITS TO THE RIGHT OF THE BINARY  
 POINT (UNSIGNED) ALL ONES FOR  
 FLOATING POINT.

CB	Item	Local	Used by: LOOK
	Boolean item set to 1 when a constant is being processed.		
CDCL1	Item	Express	Used by: MP, TSTIM
	Set equal CMPTR*1.		
CDCT	Item	Express	Used by: MP
	Counter which contains the total number of cards of direct code which have been encountered.		
CDI	Item	Express	Used by: LOOK, Main Program
	Contains beginning location in table containing item CARD into which to place a definition when a defined identifier is encountered.		
CDIT	Item	Local	Used by: LOOK
	When processing a defined variable, set to last location in table containing item CARD which will contain a character from the unpacked definition.		
CED	Table	Contains item: VALUE	Express Used by: MP
	Table of preset data which contains the KI values which immediately precede a unary operator.		
CHAN	Item	Express	Used by: MP
	Contains dictionary channel given to procedure which is called in from the library tape, but not called in the JOVIAL source program.		

CHAR	Item	Express	Used by: LOOK, MUJ
	STC variable containing the character being examined currently.		
CHCT	Item	Local	Used by: LOOK
	Contains the number of characters in the constant or identifier being processed.		
CHL1	Subscripted item from table DICT	Express	Used by: PDICT, IDEAL, POOL
	<p>If CLAS equals constant and SIND equal 1 and the nearest preceding non-constant entry is a subscripted item or array:          Specifies the number of constants (in this row).</p> <p>If CLAS equals item, subscripted item or array and Rengi equals 1:          Specifies the first character byte in item IDNS of the specified range values.</p> <p>If CLAS equals string:          Specifies the index increment to the next word of this entry containing beads.</p> <p>If CLAS equals table:          Specifies the number of entries in the table.</p> <p>If CLAS equals procedure:          Specifies the DICT entry number of the first variable associated with the procedure (0 = undeclared procedure).</p> <p>If CLAS equals file:          Specifies the maximum number of bits/bytes.</p> <p>If CLAS equals switch:          Specifies the DICT entry number of the associated item.</p> <p>If CLAS equals statement label and this is an undefined statement label in procedure:          Contains the DICT channel of this statement label if it is express.</p>		
CHL1M	Subscripted item from table DICTM	Express	Used by: PDICT, IDEAL, POOL, Main Program
	Parallel to CHL1 in the dictionary.		



CLAS	Subscripted item from table DICT	Express	Used by:	PDICT, IDEAL, POOL
	Specifies the class of the entry:			
	0 = null		7 = table	
	1 = statement label		8 = array	
	2 = file		9 = item switch	
	3 = constant		10 = numeric switch	
	4 = simple item		11 = close	
	5 = subscripted item		12 = compool program	
	6 = procedure/function		13 = string variable	
CLASM	Subscripted item from table DICTM	Express	Used by:	IMP, IDEAL
	Parallel to CLAS in the dictionary.			
CLASS	Subscripted item from table BCE	Express	Used by:	MP
	For each part of speech in BCE, CLASS contains the value which goes in CATC.			
CLASS	Subscripted item	Local	Used by:	PDICT
	STC variable in an unnamed table containing preset data, which gives a mnemonic name for each possible value of the DICT item CLAS.			
CLSIN	Item	Express	Used by:	MP
	Set by the control program to the number of columns read in on each card.			
CLSS	Output Parameter	Local	Used by:	IDEAL
	Contains the CLAS of the dictionary item.			
CMPOL	Item	Express	Used by:	IDEAL
	Boolean variable which, if true, causes the Generator to search the compool for undefined identifiers. If false, the Generator will assume no compool was available.			
CMPTR	Item	Express	Used by:	MP
	Set by the control program associated with Generator I on a particular machine. It contains the number of card columns of meaningful information on each input card.			
COUNT	Item	Local	Used by:	MUV
	Tells which half of IDNT is being used for the identifier or constant being moved.			

DAT	Item	Local	Used by: CAN
	A local variable set to the value of TYPV (dictionary item) for this constant.		
DB	Item	Local	Used by: LOOK
	Boolean item set to 1 when a dual item is being processed.		
DEBUG	File	Express	Used by: MP, PDICT, ERROR, RECRD, TSTIM
	File which contains the listing of the JOVIAL source program, error messages, CAT printout and DICT printout.		
DECOR	Switch	Local	Used by: IDEAL
	Switch set to input by IDEAL. Used to test for certain declarators by looking at first 5 characters of label in IDNT.		
	0 = 5T (SWITC)	6 = 5T (FILE)	
	1 = 5T (PROC)	7 = 5T (TABLE)	
	2 = 5T (CLOSE)	8 = 5T (ITEM)	
	3 = 5T (DIREC)	9 = 5T (MODE)	
	4 = 5T (OVERL)	10 = 5T (STRIN)	
	5 = 5T (ARRAY)		
DEF	Subscripted item	Local	Used by: PDICT
	STC variable in an unnamed table containing preset data which gives a mnemonic name for each possible value of the DICT item DEFN.		
DEFBP	Item	Express	Used by: MP, LOOK, TSTIM
	Boolean variable set to 1 when defined information is being processed.		
DEFN	Subscripted item from table DICT	Express	Used by: PDICT, IDEAL, POOL
	Specifies the definition of the entry:		
	0 = null	3 = local	
	1 = express	4 = formal procedure input parameter	
	2 = compool	5 = formal procedure output parameter	
DEFNM	Subscripted item from table DICT	Express	Used by: IDEAL
	Identical to DEFN in DICT.		

**DELIM**      Subscripted item from table BCE      Express      Used by:    MP, FNRC,  
IDEAL

Contains the last 5 out of 6 STC characters in each part of speech in table BCE.

**DENT**      Subscripted item from table DISH      Express      Used by:    SRCH, IDEAL,  
POOL

This contains the dictionary entry index. In effect, this has the dictionary channel referenced by a particular identifier (label).

**DFB**      Item      Express      Used by:    LOOK,  
Main program

Boolean item set to 1 when a DEFINE declaration is being processed.

**DI**      Item      Express      Used by:    LOOK  
Main program

Contains next available byte in variable KD.

**DICT**      Table      Contains items:    CHL1, CHL2, RNGI, RORT, TREGN, DEFN, TYPV,  
BRGT, PDAT, FCHB, TRFM, SIND, CLAS, DIMN,  
OLAY, FPNM, MAXN, MINN, PACK, FBIT, NCHB,  
SIZE

Express      Used by:    IDEAL, POOL

Contains complete description of the constants, tables, items, etc. in the JOVIAL source program as derived from their declarations.

**DICTM**      Table      Contains items:    CHL1M, CHL2M, RNGIM, RORTM, TREGNM, DEFNM,  
TYPVM, BRGTM, PDATM, FCHGM, TFRMM, SINDM,  
CLASM, DIMNM, OLAYM, FPNMM, MAXNM, MINNM,  
PACKM, FBITM, NCHEM, SIZEM

Express      Used by:    IDEAL

Parallel working storage for DICT. DICT has a variable number of entries. DICTM, however, has only four entries. Certain declarations are processed using DICTM prior to transferring to DICT. The first entry is reversed for mode, the second is for constants. In case a mode declaration is made and has parameter values, the parameters are in the second entry and the mode in the first. The third entry may act as temporary storage for simple or subscripted variables while processing a declaration. The fourth is used if the third contains a variable having parameter values, and those parameters are then set here.

DIF	<p>Item</p> <p style="text-align: right;">Local      Used by:    CAN</p> <p>This contains the number of digits between the decimal point and the end of the numeric portion of the constant. (As in 2.3E<sup>4</sup> which is changed to 23E<sup>5</sup>, DIF = 1).</p>
DIMN	<p>Subscripted item from table DICT      Express      Used by:    IDEAL, POOL</p> <p>If CLAS equals file:            Specifies the number of characters in item IDNS of the hardware label.</p> <p>If CLAS equals constant and SIND equals 1 and the nearest non-constant preceding entry is array:            Specifies the index (relative to 0) of the plane (3rd dimension) of the set of data.</p> <p>If CLAS equals array:            Specifies the number of array dimensions.</p> <p>If CLAS equals subscripted item or string:            Specifies the DICT entry number of the associated table.</p>
DIMNM	<p>Subscripted item from table DICT      Express      Used by:    IMP</p> <p>Identical to DIMN in the dictionary.</p>
DIMZ	<p>Subscripted item from table DISH      Express      Used by:    IDEAL</p> <p>For a dictionary entry with CLAS equal to 8 (ARRAY), dimension sizes of the ARRAY are stored consecutively in DISH. Reference to DIMZ is in the DICT item CHL2. The number of consecutive entries is contained in the DICT item DIMN.</p>
DIRCT	<p>File</p> <p style="text-align: right;">Express      Used by:    Main program</p> <p>Contains cards of direct code, which are output as they are encountered in the JOVIAL source program.</p>
DISH	<p>Table      Contains items:    INUS, DIMZ, SOCN, DENT</p> <p style="text-align: right;">Express      Used by:    SRCH, IDEAL</p> <p>The dictionary search table is used for entry into the dictionary. It is a one word entry table. The bottom portion of DISH is reserved for DICT overflow.</p>

**DMAIN**      Subscripted item from table **LEGAL**      Express      Used by:      **IDEAL**  
An item in table **LEGAL**, **DMAIN** is used in identification of identifiers to see if they are indeed the same. If the identifiers are within the same scope, they are then checked again to see if the identifiers are within the same range.

**DOT**      Item      Local      Used by:      **CAN**  
If item **DOT** is equal to zero, no decimal point appears in the constant. If **DOT** is not zero, it contains the actual entry of the decimal point. This is used twice with dual constants and reset between processing the halves.

**DTOB**      Output Parameter      Local      Used by:      **DTOB**  
Variable containing the output of the function **DTOB**.

**DTOB**      Procedure  
Procedure which converts an **STC** integer to binary.

**DUL**      Item      Local      Used by:      **CAN**  
A Boolean variable set to 1 if a dual constant is being processed.

**DUN**      Item      Local      Used by:      **CAN**  
A Boolean item used when processing dual constants. It is set to 1 when processing has been completed.



EFIND	Item	Express	Used by: MP, TSTIM
	Boolean variable set to 1 when an EOF is encountered during input.		
ENTER	Procedure		
	Procedure which calls IDEAL.		
ERR	Item	Local	Used by: CAN
	A Boolean variable which is set to one if an error has been found.		
ERROR	Procedure		
	Procedure which prints the error messages.		
ERT	Item	Express	Used by: ERROR
	STC parameter item which contains the standard portion of the error message printout.		
ERTY	Item	Express	Used by: ERROR
	STC item which contains output image for error messages.		
EXCID	Item	Express	Used by: MP, ENTER
	Boolean variable set to 1 when a call to procedure ENTER is made.		
EXES	Item	Local	Used by: IDEAL
	Integer counter for excessive errors. It is also used for temporary storage.		
EXTRA	Subscripted item from table BCE	Express	Used by: MP, FNRC
	Contains the first of 6 STC characters in each part of speech in table BCE.		

FB	Item	Express	Used by: LOOK, MUV
	Boolean item set to 1 the first time procedure MUV is called from LOOK, each time LOOK is used.		
FBIT	Subscripted item from table DICT	Express	Used by: PDICT, IDEAL, POOL
	If CLAS equals subscripted item and the item is in a table with specified packing: Specifies the first bit.		
FBITM	Subscripted item from table DICTM	Express	Used by: IMP
	Equivalent to FBIT in DICT.		
FCHAR	Item	Local	Used by: LOOK
	STC item used to hold one character when scanning a comment to discard it.		
FCHB	Subscripted item from table DICT	Express	Used by: ERROR, PDICT, SRCH, IDEAL, POOL
	Specifies the first character byte in item IDNS of the identifier associated with the entry.		
FCHBM	Subscripted item from table DICTM	Express	Used by: IDEAL, IMP
	Parallel to FCHB in DICT.		
FIND	Table	Contains item: BIND	Express Used by: MP
	Table used to store counts of parentheses when processing nested functional modifiers.		
FIRST	Item	Express	Used by: IDEAL
	An item containing the first dictionary index to be printed.		
FIRST	Input Parameter	Local	Used by: PDICT
	Contains the first channel of the dictionary (DICT) to be edited for printing.		
FNRC	Procedure		
	Procedure which provides for restarting operation after an illegal triplet has been detected.		

<b>FOND</b>	Item	Express	Used by:	<b>HASH, SRCH IDEAL, POOL</b>
	Boolean item used as communication between IDEAL and SRCH. If FOND is set to 1 on entering SRCH, SRCH has already been entered for this item and now overflow is to be checked. If FOND is set to 1 upon returning to IDEAL, an identifier identical to the one being processed is already in the dictionary.			
<b>FORM</b>	Item in table BCE	Express	Used by:	<b>MP</b>
	For each part of speech in BCE, FORM contains the value which goes in CATF.			
<b>FPNM</b>	Subscripted item from table DICT	Express	Used by:	<b>PDICT, IDEAL, POOL</b>
	If CLAS equals procedure; Specifies the number of formal parameters.			
	If CLAS equals subscripted item or a string variable in a table with specified packing: Specifies the word of the entry for the item or string.			
	If CLAS equals table, array, item, statement label, close, numeric switch, item switch, or file and DEFN greater than 3: Specifies the formal parameter number.			
<b>FPNM</b>	Subscripted item from table DICTM	Express	Used by:	<b>IDEAL, IMP</b>
	Identical to FPNM in the dictionary, formal parameter number.			

GET2      Procedure  
          Procedure which calls LOOK to bring the next part of speech into  
          IDNT.

GET3      Procedure  
          Procedure which calls LOOK until a dollar sign is detected.

GT        Switch                      Local      Used by:    LOOK  
          Entered when a special character has been recognized. The entrance  
          value is obtained from subscripted item SV.

HASH	Procedure		
	Procedure which converts a string of STC characters to an index value.		
HB	Item	Local	Used by: LOOK
	Boolean item set to 1 when a Hollerith or STC constant is being processed.		
HI	Item	Express	Used by: MP
	Used for temporary storage during processing of direct code.		
HIN	File	Express	Used by TSTIM, Main program
	File containing the Hollerith input to the Generator, i.e., the JOVIAL source program.		
HL	Item	Local	Used by: LOOK
	Contains the number of characters + 2 in a Hollerith or STC constant when processing the constant.		

II	Item	Express	Used by: MP, ENTER, LOOK, RECRD, TSTIM
	The index to CAT.		
IDA	Subscripted item from table IDT	Express	Used by: MUV
	Item overlayed with one character in IDNT, to provide access to characters without using the string routines.		
IDB	Subscripted item from table IDT	Express	Used by: MUV
	Item overlayed with one character in IDNT, to provide access to characters without using the string routines.		
IDC	Subscripted item from table IDT	Express	Used by: MUV
	Item overlayed with one character in IDNT, to provide access to characters without using the string routines.		
IDD	Subscripted item from table IDT	Express	Used by: MUV
	Item overlayed with one character in IDNT, to provide access to characters without using the string routines.		
IDE	Subscripted item from table IDT	Express	Used by: MUV
	Item overlayed with one character in IDNT, to provide access to characters without using the string routines.		
IDEAL	Procedure		
	Procedure which processes certain declarations, totally or partially, invoking the aid of specific processing procedures where necessary. It makes dictionary (DICT) entries defining labels and constants, processes preset parameter constants included with declarations, and stores identifiers and constants in IDNS.		
IDF	Subscripted item from table IDT	Express	Used by: MUV
	Item overlayed with one character in IDNT, to provide access to characters without using the string routines.		
IDG	Subscripted item from table IDT	Express	Used by: MUV
	Item overlayed with one character in IDNT, to provide access to characters without using the string routines.		
IDH	Subscripted item from table IDT	Express	Used by: MUV
	Item overlayed with one character in IDNT, to provide access to characters without using the string routines.		

IDN	Item	Express	Used by: MP
	Dummy item used to overlay the NENT word of table IDT.		
IDNS	Item	Express	Used by: IMP, SRCH, PDICT, POOL, ERROR, IDEAL
	STC variable which contains all identifiers and constants found in the JOVIAL source program.		
IDNT	Item	Express	Used by: LOOK, BINRY, HASH, ERROR, FMRC, IDEAL, POOL, CAN, SCON, SRCH, GET2, IMP, IDEAL
	STC variable which contains the current part of speech being processed, and the last part of speech when the part of speech is an identifier or constant.		
IDOL	Item	Express	Used by: MP, GET2, GET3, IDEAL
	A Boolean variable set to 1 if IDEAL uses LOOK to move past the end of a declaration.		
IDT	Table	Contains items IDA, IDB, IDC, IDD, IDE, IDF, IDG, IDH	
		Express	Used by: MUV
	Overlayed with item IDNT to provide access to characters without using the string routines.		
IM	Item	Local	Used by: CAN
	A buffer string that holds the 128 characters being processed and is needed in order to maintain the original value in IDNT. One character at a time is scanned and placed in IM; when completed this entire entry is placed back into IDNT.		
IMP	Procedure		
	Procedure which processes that portion of declarations called the item description including the preset parameter constant if it is a single constant in a non-subscripted variable declaration.		
INC	Item	Local	Used by: LOOK
	A Boolean variable set to 1 if the index to CARD (J1) is to be incremented by 1 before return from LOOK, and 0 if not.		

10 July 1962

32  
Last Page

TM-555/020/00C

INDEX	Input Parameter	Local	Used by: SCON
	Contains a STAT table index if a search is requested.		
INDIR	Item	Express	Used by: MP
	A Boolean variable set to 1 when processing is being done inside of DIRECT - JOVIAL brackets.		
INITBYTE	Item	Express	Used by: Main program
	Contains byte in TSTR into which the first character of the input is read.		
INUS	Subscripted item from table DISH	Express	Used by: SRCH, POOL, IDEAL
	A Boolean variable in DISH table. 1 equals true which implies that this DISH channel is occupied.		



10 July 1962

33

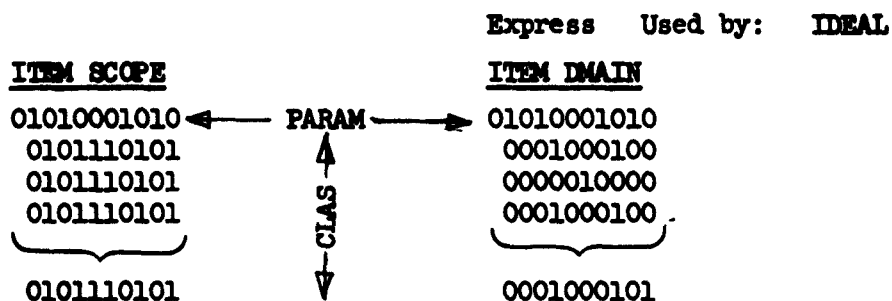
TM-555/020/00

J1	Item	Express	Used by:	MP, LOOK, RECRD, TSTIM
	Index to IMAGE.			
JJ	Item	Express	Used by:	LOOK, MP
	Holds level of nesting when a defined identifier is being processed.			
JUNK	Item	Express	Used by:	MP
	Used to discard output from LOOK which is not needed.			

K	Subscript	Local	Used by: IDEAL
	Counter used in reference to DICT; it holds the number of the dictionary channel that is in use.		
K1	Item	Express	Used by: MP, FNRC, ENTER
	Contains value associated with the left term of the triplet being processed currently.		
KD	Item	Express	Used by: LOOK
	STC item which holds defined identifiers and their definitions (from DEFINE declarations).		
KDA	Subscripted item from unnamed table	Express	Used by: LOOK
	Contains first character byte of defined identifier from DEFINE declaration.		
KDB	Subscripted item from unnamed table	Express	Used by: LOOK
	Contains number of characters in defined identifier from DEFINE declaration		
KDC	Subscripted item from unnamed table	Express	Used by: LOOK
	Contains number of characters in definition from DEFINE declaration.		
KVAL	Subscripted item from table BCE	Express	Used by: MP, FNRC
	Contains the K1 value for each part of speech in table BCE.		

L1	Item	Express	Used by: FARC, GET2, GET3, IMP, IDEAL, LOOK, TSTIM, MP
	Contains value associated with the right term of the triplet being processed currently.		
L1	Item	Local	Used by: CAN
	Used as a pointer to reference the position with IDNS string for the constant being analyzed, left pointer.		
L2	Item	Local	Used by: CAN
	A local identifier used as a left reference point to the constant image.		
L3	Item	Local	Used by: CAN
	Used as a temporary storage for left pointer in processing dual items.		
LAST	Item	Express	Used by: IDEAL
	Contains the most recent dictionary (DICT) channel reference for a status variable.		
LAST	Input Parameter	Local	Used by: PDICT
	Contains the last channel of the dictionary (DICT) to be edited for printing.		
LB	Item	Local	Used by: LOOK
	Boolean item set to 1 when the first letter in an identifier, or the first number in a constant is encountered.		
LEB	File	Express	Used by: TSTIM, MP
	File which holds the JOVIAL library procedures.		
LEBTP	Subscripted item in control table	Express	Used by: MP
	Boolean item set to 1 when a library tape is available and set to 0 otherwise.		
LEBIN	Item	Express	Used by: TSTIM, MP
	Boolean item set to 1 when processing of the library procedures begins.		
LEFN	Item	Express	Used by: TSTIM, MP
	Boolean item set to 1 when an end of file is detected on the library tape.		

LCHAR	Item	Local	Used by:	LOOK
	STC item which contains the last character processed.			
LCRD	Item	Express	Used by:	MP
	Holds number of cards in a library procedure when that procedure is to be bypassed on the library tape.			
LEGAL	Table	Contains items: SCOPE, DMAIN		



If the identifier is already entered in the dictionary, SCOPE and DMAIN are used to test for usage to see if the identifiers are actually the same.

(CLAS[ID] \* 10 + PARAM = denotes index to SCOPE)

LOOK	Procedure
	Procedure which finds the next part of speech in the JOVIAL source program.

**TM-555/020/00**

\* 709 VERSION OF LEGALITY MATRIX

	CIS	*
	ODU	*
	= . (\$ / * INEB	*
	, , \$ = . ( ) \$ / ) - + * / * I SNS	*
	1234567890123456789012	*
BCI	4,999999999999999999999999	NULL 1
BCI	4,9191999919999119999911199	DESL 2
BCI	4,991699314999111119911199	FILL 3
BCI	4,911111119199111119911199	CONS 4
BCI	4,911111119199111119911191	ITEL 5
BCI	4,911191119199111119911199	SITL 6
BCI	4,991999119999119999911199	PROL 7
BCI	4,91111119199111119911191	SUBS 8
BCI	4,999299794999662229916699	IF 9
BCI	4,999599995999999999999999	GOTO 10
BCI	4,992999999999999999999999	FOR 11
BCI	4,9999999999999999999991199	RETURN 12
BCI	4,9995999999999999999991199	STOP 13
BCI	4,9992999999999999999991199	TEST 14
BCI	4,592729794999119199916699	. 15
BCI	4,562699764299662229911199	, 16
BCI	4,569299724999662229916199	= 17
BCI	4,999299394999999999999999	== 18
BCI	4,999999724699666669911199	... 19
BCI	4,962196764199666669916699	( 20
BCI	4,911191119199111119911191	) 21
BCI	4,929992794299662229911199	(\$ 22
BCI	4,911111119199111119911191	\$) 23
BCI	4,999599999999999999999999	TERM 24
BCI	4,999299724999662229916699	LOGO-RELO 25
BCI	4,999299724999662229916699	RELO 26
BCI	4,922292724299662229916699	ARIO 27
BCI	4,999999199999119999911199	BEAD 28
BCI	4,999999199999999999911199	FMOD 29
BCI	4,999299794999662229916699	IFEITH 30
BCI	4,999299794999662229916699	ORIF 31
BCI	4,592729794999119199916699	BEGIN 32
BCI	4,592729794999119199916699	END 33
BCI	4,592729794999119999916699	START 34
BCI	4,592729794999119199916699	\$ 35
BCI	4,592129394999119999911199	DIRECT 36
BCI	4,999999299999229999992999	INPUT 37
BCI	4,999999299999229999992999	OUTPUT 38
BCI	4,999299299999229999992999	O/S IN 39
BCI	4,999299299999229999992999	O/S OU 40

MATI	Subscripted item from table MAT	Express	Used by: MP
	STC item which contains the values in the legality matrix.		
MAXN	Subscripted item from table DICT	Express	Used by: IDEAL
	Overlays the least significant bits of DIMN and may be referenced for the number of characters in the hardware label of a file.		
MAXNM	Subscripted item from table DICTM	Express	Used by: Not used
MINN	Subscripted item from table DICT	Express	Used by: Not used
MINNM	Subscripted item from table DICTM		Used by: Not used
MORED	Item	Express	Used by: MP
	Boolean item set to 1 if DIRECT is the third term in the triplet being processed.		
MTRX	Item	Express	Used by: MP
	Transmission code variable which is overlayed with MAT.		
MTYP	Item	Express	Used by: MP
	Set to 1, 2, or 3 when the values 6, 7, or 8 respectively are encountered in the legality matrix.		
MUV	Procedure		
	Procedure which moves one character of information from item CHAR to the next available byte in IDNT.		

NARDA	Item	Express	Used by:	IDEAL, POOL
	Counter for the next available channel containing DIMZ in the DISH table.			
NBYT	Item	Express	Used by:	Main program
	Parameter set to the number of bytes in a machine word (machine dependent).			
NC	Item	Express	Used by:	LOOK
	Parameter item set to maximum allowable entry for subscripted item CARD.			
NCHAR	Item	Express	Used by:	MP, FNRC, CAN, ENTER, POOL, HASH, GET3, SCON, SRCH, GET2, IMP, IDEAL
	Contains the number of characters in the middle term of the triplet being processed currently.			
NCHB	Subscripted item from table DICT	Express	Used by:	ERROR, PDICT, IDEAL, POOL, SRCH
	If CLAS equals constant and SIND equals 1 and the nearest preceding non-constant entry is an item or array: Specifies the number of bytes in the first constant set (excluding the separator character). For all other classes: Specifies the number of character bytes in item IDNS of the identifier associated with the entry.			
NCHEM	Subscripted item from table DICTM	Express	Used by:	IMP, IDEAL
	Parallel to NCHB, NCHEM is storage and reference when making up declaration items for the dictionary.			
NCT	Item	Express	Used by:	Main program
	Parameter item set to maximum allowable entry in table CAT.			
ND	Item	Express	Used by:	LOOK
	Parameter item set to maximum allowable entry in table DICT.			
NDEFS	Item	Express	Used by:	MP, LOOK
	Contains the number of DEFINE declarations processed.			

NDICT	Item	Express	Used by:	IDEAL, IMP POOL
	Counter used in reference to DICT; it holds the number of the next unused dictionary slot. At any given time NDICT -1 is the number of DICT entries. DICT is filled in order of increasing magnitude with the zeroeth entry unused.			
NDICTM	Item	Express	Used by:	IDEAL
	Index into DICTM.			
NEG	Item	Local	Used by:	CAN
	A Boolean variable set to 1 if a negative value is being processed. It is used twice in CAN; once at the beginning for prefixed sign and again for $\pm$ A or E factor.			
NEXT	Item	Express	Used by:	IDEAL, POOL, LOOK, HASH, PDICT, SRCH, IMP, MP
	Used as a counter for IDNS. At any given time this variable contains the next unused byte in IDNS. IDNS bytes are filled in order of increasing magnitude.			
NI	Item	Express	Used by:	LOOK
	Parameter item set to maximum allowable byte in item IDNS.			
NO	Item	Express	Used by:	LOOK
	Parameter item set to maximum allowable entry in table OVER.			
NOCN	Item	Express	Used by:	MP
	Counter used to save the number of direct code statements between (1) DIRECT and JOVIAL, (2) DIRECT and ASSIGN, (3) ASSIGN and ASSIGN, (4) ASSIGN and JOVIAL brackets.			
NOCN	Item	Express	Used by:	SRCH, IDEAL
	Contains the next overflow channel available in table DISH: counter used to store SOCN. Initial value is 1025.			
NORI	Item	Express	Used by:	MP, ENTER
	Boolean variable set to 1 when an item switch is being processed.			
NOVER	Item	Express	Used by:	IDEAL, LOOK
	Counter for entry into the table OVER.			



18 September 1962

41

TM-555/020/008

NSIAT	Item	Express	Used by: SCON, IDEAL, POOL
	Contains the next available slot in the STAT table. NSIAT-1 is the number of status variables in the source program.		
NU	Item	Express	Used by: LOOK
	Parameter item set to maximum allowable number of define declarations.		
NUMB	Input Parameter	Local	Used by: IDEAL
	Input from ENTER to IDEAL, NUMB contains the number of characters to be processed.		
NUMB	Input Parameter	Local	Used by: BINRY
	Contains the number of characters in IDNT to be converted.		
NVAL	Subscripted item from table STAT	Express	Used by: SCON, IMP, IDEAL
	Contains the number of stati connected with this particular variable.		
NVALU	Item	Express	Used by: SCON, IDEAL, POOL
	Counter for entry into the status constant string, VALU; it gives the next available byte for storage.		
NX	Item	Express	Used by: LOOK
	Parameter item set to maximum byte allowable in item KD. (Maximum number of characters allowable in total of defined labels and their definitions.)		
NXTCR	Item	Express	Used by: MP, FNRC, GET2, GET3
	Contains the number of characters in the right term of the triplet being processed currently.		

OA Subscripted item from table OV Express Used by: TSTIM  
STC item in table overlayed with STC item TSTR, to provide access to characters without using the string routines.

OB Item Express Used by: MP  
Dummy item used to overlay the NENT word of table OV.

OCT Item Local Used by: CAN  
An octal 63, it is used with SEP as a separator between entries in IDNS.

OLAY Subscripted item from table DICT Express Used by: CAN  
If CLAS equals item, table, array or file:  
0 = not used in an overlay  
1 = used in an overlay

OLAYM Subscripted item from table DICTM Express Used by: Not used  
Identical to OLAY in the dictionary.

ORGY Subscripted item from table OVER Express Used by: Not used  
This is the overlay guaranty storage and is available to the translator in the packing phase.

OSB Item Local Used by: LOOK  
Boolean item set to 1 when an octal or status constant is being processed.

OUT Item Express Used by: RECRD  
STC item which contains output image for CAT printout.

OV Table containing item: OA Express Used by: TSTIM  
Table overlayed with STC variable TSTR to provide access to characters without using the string routines.

OVEN Subscripted item from table OVER Express Used by: IDEAL  
This is the overlay sequence number and starts at 1 for each overlay declaration.

OVER Table Contains items: OVID, OVEN, ORGY, OVLI  
Express Used by: IDEAL  
An ancillary table containing OVERLAY declaration information.

10 July 1962

43

TM-555/020/00

OVID      Subscripted item from table OVER      Express      Used by:      IDEAL  
Contains DICT channel of the variable.

OVLI      Subscripted item from table OVER      Express      Used by:      IDEAL  
Overlay indicator. If equal to 0, this is the lead of a series of  
variables. If 1, the variables are consecutive.

As in the case:

OVERLAY AA = BB = CC (OVLI is set to 0 in all three)  
OVERLAY AA, BB = CC, DD (OVLI equals 0, 1, 0, 1 in that order  
of succession)

PAC	Subscripted Item	Local	Used by: PDICT												
STC variable in an unnamed table containing preset data which gives a mnemonic name for each possible value of the DICT item PACK.															
PACK	Subscripted item from table DICT	Express	Used by: PDICT, IDEAL POOL												
Specifies the packing of a subscripted item or table.															
<table border="0"> <tr> <td>0 = null</td> <td>3 = medium (M)</td> </tr> <tr> <td>1 = specified</td> <td>4 = dense (D)</td> </tr> <tr> <td>2 = loose or none (N)</td> <td></td> </tr> </table>				0 = null	3 = medium (M)	1 = specified	4 = dense (D)	2 = loose or none (N)							
0 = null	3 = medium (M)														
1 = specified	4 = dense (D)														
2 = loose or none (N)															
PACKM	Subscripted item from table DICTM	Express	Used by: IMP, IDEAL												
Identical to PACK in the dictionary.															
PARAM	Item	Express	Used by: MP, ENTER, IDEAL, POOL												
Set for input to IDEAL gives reason for entry.															
<table border="0"> <tr> <td>0 = Housekeep</td> <td>6 = Constant</td> </tr> <tr> <td>1 = Designation label</td> <td>7 = Switch label</td> </tr> <tr> <td>2 = Procedure label</td> <td>8 = Switch item</td> </tr> <tr> <td>3 = Statement label</td> <td>9 = Close label</td> </tr> <tr> <td>4 = Variable</td> <td>10 = Modified variable</td> </tr> <tr> <td>5 = Subscripted item</td> <td>11 = Terminal entry to IDEAL</td> </tr> </table>				0 = Housekeep	6 = Constant	1 = Designation label	7 = Switch label	2 = Procedure label	8 = Switch item	3 = Statement label	9 = Close label	4 = Variable	10 = Modified variable	5 = Subscripted item	11 = Terminal entry to IDEAL
0 = Housekeep	6 = Constant														
1 = Designation label	7 = Switch label														
2 = Procedure label	8 = Switch item														
3 = Statement label	9 = Close label														
4 = Variable	10 = Modified variable														
5 = Subscripted item	11 = Terminal entry to IDEAL														
PB	Item	Local	Used by: LOOK												
Boolean item used when processing a dual constant. Set to 1 when a left parenthesis is encountered inside the constant, and set to 0 when a right parenthesis is encountered inside the constant.															
PCAT1	Item	Express	Used by: RECRD												
Boolean variable set by the control program as follows:															
<table border="0"> <tr> <td>1 = CAT table to be printed</td> </tr> <tr> <td>0 = CAT table not to be printed</td> </tr> </table>				1 = CAT table to be printed	0 = CAT table not to be printed										
1 = CAT table to be printed															
0 = CAT table not to be printed															
PCAT2	Item	Express	Used by: MP												
Boolean variable set by control program which, if true, indicates that the CAT table should be printed by Phase II.															
PCHN	Item	Express	Used by: IDEAL, POOL												
Temporary storage containing the procedure channel number. If a procedure declaration is being processed, this is set as an index to the procedure identifier.															

**PDAT**      Subscripted item from table DICT      Express      Used by:    PDICT, IMP,  
CAN, IDEAL,  
POOL

    If CLAS equals file:  
        0 = fixed length records  
        1 = variable length records

    If CLAS equals table:  
        0 = fixed length  
        1 = variable length

    If CLAS equals item, subscripted item or array and TYPV equals  
A, D, F, I or K:  
        0 = unsigned  
        1 = signed

    If CLAS equals procedure:  
        0 = express items are set in the procedure  
        1 = no express items are set in the procedure

**PDATM**      Subscripted item from table DICTM      Express      Used by:    IMP, IDEAL,  
CAN

    Equivalent to PDAT.

**PDCTY**      Subscripted item in control table      Express      Used by:    PDICT

    Boolean item set to 1 by the control program when the dictionary is  
to be printed.

**PDICT**      Procedure

    Procedure which prints the identifier storage item, IDNS, and the  
dictionary, DICT.

**PFB**      Item      Local      Used by:    LOOK

    Boolean item set to 1 when a period is encountered followed by a  
number

**POOL**      Procedure .

    Procedure which extracts certain definitions from the compool and  
places them in the dictionary (DICT).

**PP**      Item      Express      Used by:    LOOK  
Main program

    Contains number of undeclared procedures in the JOVIAL source program.

PREP	Item	Express	Used by:	MP, ERROR, IDEAL, POOL
	Boolean variable set to 1 when a procedure is being processed.			
PRCD	Item	Express	Used by:	MP, ERROR
	Item set to the channel of the name of the procedure being processed.			
PRERI	Item	Express	Used by:	MP
	Boolean variable set to 1 when a procedure declaration within a procedure declaration is found.			
PRSC	Item	Express	Used by:	MP, ERROR, LOOK
	Counter used for the number of statements in a procedure since the last label.			
PRST	Item	Express	Used by:	MP, ERROR
	Contains the channel number of the last label encountered within a procedure.			
PSSL	Item	Express	Used by:	MP
	Boolean variable set to 1 when a statement label is used as a parameter in a procedure declaration.			
PTYP	Item	Express	Used by:	MP, IDEAL
	Item which gives parameter type when processing a procedure declaration.			
	0 = no setting 1 = formal input parameter 2 = formal output parameter			
PURE	Table containing items URE1, URE2	Express	Used by:	MP
	Holds names of undeclared procedures.			

R1	Item	Local	Used by:	CAN
	Used as a right reference pointer for IDNT, this will never be less than L1. Necessary to know the reference to the constant being analyzed at any time, character referenced.			
R2	Item	Local	Used by:	CAN
	A right reference pointer for the image string, character reference for position at any time during the constant analysis.			
R3	Item	Local	Used by:	CAN
	Temporary storage while processing dual constants, contains the right limits.			
R4	Item	Local	Used by:	CAN
	Temporary storage when processing a non-dual constant, contains the extreme right limits.			
RECRD	Procedure			
	Procedure which prints the CAT table.			
RNGI	Subscripted item from table DICT	Express	Used by:	IDEAL
	If CLAS equals item, subscripted item or array: 0 = no range specified 1 = range specified			
RNGIM	Subscripted item from table DICTM	Express	Used by:	IMP
	Identical to item RNGI in the dictionary.			
RORT	If CLAS equals item, subscripted item or array: 1 = round variable 0 = truncate variable			
RORTM	Subscripted item from table DICTM	Express	Used by:	IMP
	Identical to Boolean variable in the dictionary, information as whether to truncate or round.			
RSON	Input Parameter	Local	Used by:	IDEAL
	Contains the value of PARAM from main program.			
RTPAR	Item	Express	Used by:	MP
	Boolean variable set to 1 when the next right parenthesis is to be eliminated in the CAT table.			

S1	Switch	Express	Used by: MP
	Switch entered with the value picked up from the legality matrix.		
S2	Switch	Express	Used by: MP
	Switch entered when declaration has been recognized. Distinguishes between PROC, SWITCH and CLOSE.		
S3	Switch	Express	Used by: MP
	Switch entered when a recognizable part of speech has been found (one contained in table BCE).		
SAV	Subscripted item	Express	Used by: LOOK
	Item in unnamed table which contains CARD entry (J1) when processing nested defined identifiers.		
SB	Item	Local	Used by: LOOK
	Boolean item set to 1 when a letter is the first character in a part of speech.		
SC	Subscripted item	Local	Used by: LOOK
	STC item in unnamed table. Each entry contains one JOVIAL character (may be 2 bytes) that is recognized by LOOK		
SCON	Procedure		
	Procedure which stores stati for variable declarations having stati, finds stati assigned to a variable among that variable's associated stati, and converts to an integer in STC if found.		
SCON	Output Parameter	Local	Used by: SCON
	A Boolean item set for immediate testing on exit. If true, a status was found and converted. If false, the search for status in a given channel of the STAT table was unsuccessful.		
SCOPE	Subscripted item from table LEGAL	Express	Used by: IDEAL
	Used to see whether an identifier is within the same scope or usage as another identifier having the same name. Used in conjunction with DMAIN.		
SDT	Item	Local	Used by: CAN
	A local variable set to the present value of PDAT (dictionary item).		



SEP	Item	Local	Used by:	CAN
	A local variable which stands for separator and is used in conjunction with OCT to separate IDNS entries.			
SETK	Table	Contains item: VALK	Express	Used by: MP
	Contains preset data used for setting the value of K1, the left term in a triplet.			
SGN	Item	Local	Used by:	CAN
	Boolean item set to true when the ITEM being processed is signed.			
SIND	Subscripted item from table DICT	Express	Used by:	PDICT, IDEAL
	If CLAS equals file: 0 = Hollerith 1 = binary			
	If CLAS equals procedure: 0 = non-function 1 = function			
	If CLAS equals constant: 0 = is not initial data 1 = initial data			
	If CLAS equals table, item, subscripted item or array: 0 = no initial data 1 = variable has initial data			
	If CLAS equals statement label or close: 0 = the label/close was used after its declaration (at least one backward transfer) or else it was used in a switch. 1 = label/close was always used before its declaration (no backward transfers)			
SINDM	Subscripted item from table DICTM	Express	Used by:	IMP, IDEAL
	This is equivalent to SIND in the dictionary; CLAS dependent.			

**SIZE**      Subscripted item from table DICT      Express      Used by:      PDICT, IDEAL  
POOL

If CLAS equals table and PACK equals 1:  
    Specifies the number of words per entry.

If CLAS equals item, subscripted item or array:  
    (1) TYPV equals H, T:  
        Specifies the number of bytes.  
    (2) TYPV equals A, D, I, S, K:  
        Specifies the number of bits (including the sign bit)

If CLAS equals file:  
    Specifies the maximum number of records.

If CLAS equals constant and SIND equals 1 and the nearest non-constant preceding entry is an array:  
    Specifies the index to the assigned column (relative to 0) for this constant set.

Note: If a procedure is being processed and it is a function, SIZE contains the DICT channel of the output parameter.

**SIZEM**      Subscripted item from table DICTM      Express      Used by:      IMP, IDEAL  
Equivalent to SIZE which is an item in the dictionary.

**SOCN**      Subscripted item from table DISH      Express      Used by:      SRCH  
Index to overflow portion of DISH table. A setting of zero indicates that no overflow exists. Overflow occurs in two cases:  
    (1) the same identifier references different variables  
    (2) different identifiers hash to the same number  
Hash refers to a method of chopping an identifier to an index value for search purposes.

**SPEC**      Item      Express      Used by:      IMP, IDEAL  
This variable is used as internal communication between IMP and IDEAL for a specified table or function variable. This is a Boolean variable, set to 1 when a table that is completely specified is being processed.

**SRCH**      Procedure  
Procedure using an access table called DISH with an index value supplied by the procedure HASH to determine the presence of a label, i.e., identifier, or constant defined in the dictionary, DICT. The label being searched is in IDNT. The results of the search are communicated with a Boolean indicator, and if found, an index value to the defining DICT entry is supplied. Housekeeping for reference and entry into IDNS is supplied. DISH items are maintained.

STAT	Table	Contains items: NVAL, VPOS	Express	Used by: IDEAL
<p>This table references the status constant string, VALU. VPOS is the byte number in VALU containing the first byte of the first status label of this variable. NVAL is the number of status labels of this variable. NSTAT is the number of entries in this table. DICT entries defining status variables contain a reference to this table in BRGT.</p>				
STC	Item		Express	Used by: IDEAL, POOL
<p>Used for temporary storage for STC characters in IDEAL. Also used for communication between IDEAL and POOL when LIKE tables are being processed.</p>				
STC	Item		Local	Used by: CAN
<p>Used for storing STC characters.</p>				
STCTR	Item		Express	Used by: MP, ERROR, LOOK
<p>Counter containing the number of statements found since last statement label in the main program.</p>				
STMT	Item		Express	Used by: MP, ERROR
<p>Set to the channel number of the last statement label in the main program.</p>				
STORE	Input Parameter		Local	Used by: SCON
<p>A Boolean indicator. If STORE is true, SCON stores stati; if false, it searches STAT table for status in channel referenced by INDEX.</p>				
SUM	Item		Express	Used by: MP, IDEAL
<p>Set to the entry in table BCE of the separator, operator or bracket processed most recently.</p>				
SV	Subscripted item		Local	Used by: LOOK
<p>Item in unnamed table which gives entrance value into switch GT for each value in item SC.</p>				
SWEP	Item		Express	Used by: MP, ENTER, IDEAL
<p>Boolean variable set to 1 when SWITCH declaration is being processed.</p>				

10 July 1962

52

TM-555/020/00

SWCN	Item	Express	Used by:	IDEAL, MP
	This item is set to the switch channel number and is used as temporary storage for an index to the dictionary when a switch is being processed.			
SWDS	Item	Express	Used by:	Main program
	Boolean item set to 1 when the dollar sign terminating a switch declaration is encountered.			

T0	Item	Local	Used by:	CAN
	Temporary storage register in CAN.			
T1	Item	Local	Used by:	BINRY
	Used as an integer counter.			
T1	Item	Express	Used by:	IDEAL, GET2, GET3, IMP
	Temporary storage.			
T2	Item	Express	Used by:	CAN, IDEAL, SCON, IMP
	Used as two-way communication between CAN and IDEAL. In IDEAL, T2 is used for temporary storage as with table declarations. Also, if searching for an identifier, CLAS is stored in T2 temporarily if one is found to look like the same identifier.			
	T2 = 0 informs CAN that the constant being processed is literal			
	T2 = 1 informs CAN that the constant being processed is a parameter for a simple variable which is to be defined by the constant.			
	T2 > 1 is the DICT index of the variable for which this constant is a pre-assigned parameter.			
T3	Item	Express	Used by:	HASH, IDEAL, IMP, POOL, SRCH
	Used for temporary storage; normally as an index to IDNS.			
T4	Item	Express	Used by:	HASH, SRCH, IDEAL, POOL
	Internal index into DISH, this is the HASHed entry plus one.			
T5	Item	Express	Used by:	IDEAL, SCON, GET2, GET3, IMP, CAN, ERROR
	Temporary storage for the number of characters to be processed.			
T6	Subscripted item from table TPY	Express	Used by:	HASH
	Overlays the first 10 bits of the identifier in TH that is to be "hashed."			

T7	Subscripted item from table TPY	Express	Used by: HASH
	Overlays the second 10 bits of the identifier in TH that is to be "hashed."		
T8	Subscripted item from table TPY	Express	Used by: HASH, BINRY
	In HASH, T8 overlays the third 10 bits of the identifier in TH that is to be "hashed." In BINRY, T8 is used as temporary storage for the bytes of the identifier while it is being converted.		
T9	Subscripted item from table TPY	Express	Used by: HASH
	Accumulated sum of T6, T7, and T8. This is the "hashed" identifier.		
T10	Item	Local	Used by: BINRY
	Used to retain powers of ten in the STC to binary conversion process.		
T10	Item	Local	Used by: IDEAL
	Used for temporary storage.		
T10	Item	Express	Not Used
T11	Item	Express	Used by: IDEAL
	Used for temporary storage.		
T12	Item	Express	Used by: IDEAL, IMP, SCON
	Used for temporary storage.		
T13	Item	Express	Used by: IDEAL, POOL, IMP, SCON
	Contains the dictionary entry reserved for the variable to be defined.		
T13	Item	Local	Used by: HASH
	Used for temporary byte number storage.		
T14	Item	Express	Used by: IDEAL
	Used for temporary storage.		
T15	Item	Express	Used by: IDEAL
	Temporary storage for RSON (reason for entry).		

T16	Item	Local	Used by:	IDEAL
	Counter used when an ARRAY is being processed.			
T17	Item	Local	Used by:	IDEAL
	Counter used when an ARRAY is being processed			
TAPE	Item	Express	Used by:	PDICT
	STC variable which holds one line of the DICT output image for printing.			
TBBP	Item	Express	Used by:	IMP, IDEAL
	A Boolean variable, which is set to 1 if a table declaration is being processed.			
TCD	Item	Express	Used by:	TSTIM, Main program
	Set to entry of CARD which contains first character from JOVIAL source language card input.			
TCD1	Item	Express	Used by:	LOOK, TSTIM, Main program
	Set to entry of CARD which is 6 less than entry that contains last character from JOVIAL source language card input.			
TCD2	Item	Express	Used by:	TSTIM, Main program
	Set to entry in CARD which is 6 less than entry that contains first character from JOVIAL source language input.			
TCH	Item	Local	Used by:	LOOK
	STC item used to hold two characters to test them against special characters in item SC.			
TCHN	Item	Express	Used by:	IDEAL, IMP
	If a table is being processed, TCHN is set to the actual table channel number of TBBP. When not processing a table, TCHN is used in an ambiguous case of identifier overlap for storage purposes. When an identifier is being processed, and may be EXPRESS in the main program but also LOCAL, TCHN is then used as temporary dictionary reference storage.			

TEM1	Item	Local	Used by:	LOOK
	Temporary storage used when processing defined identifiers.			
TEM2	Item	Local	Used by:	LOOK
	Temporary storage used when processing defined identifiers.			
TEXT	Item	Express	Used by:	MP, LOOK, IDEAL, BINRY, HASH, ERROR, SCON, SRCH, GET2, IMP, CAN, POOL, FNRC, MUV, CTEXT
	Index in IDNT which is set to the location of the first byte in one half of IDNT into which the next part of speech will be placed by LOOK.			
TFR	Subscripted item	Local	Used by:	PDICT
	STC variable in an unnamed table containing preset data which gives a mnemonic name for each possible value of the DICT item TFRM.			
TFRM	Subscripted item from table DICT	Express	Used by:	PDICT, IDEAL, POOL
	If CIAS equals table:			
	0 = null			
	1 = parallel table			
	2 = serial table			
TFRM	Subscripted item from table DICTM	Express	Used by:	IDEAL
	Identical to the dictionary item TFRM.			
TH	Subscripted item from table TH	Express	Used by:	HASH, BINRY
	In HASH, the STC variable TH contains five or less bytes of the identifier that is being "hashed." In BINRY, TH is used for temporary storage for five or less bytes of the identifier that is being converted from an STC to a binary integer.			
TIPE	Item	Express	Used by:	MP, FNRC, GET2, GET3, IDEAL, SCON, IMP
	Contains the type (output from LOOK) of the part of speech currently being processed.			
TITLE	Item	Local	Used by:	PDICT
	STC variable containing preset data giving the heading printed above the dictionary entries printed by PDICT.			



TP	Item	Express	Used by:	TSTIM, Main program
	Set to 1 less than the number of bytes per word.			
TP1	Item	Express	Used by:	TSTIM, Main program
	Set to 1 less than the number of words per card image.			
TPY	Table	Contains items: TH, T6, T7, T8, T9		
		Express	Used by:	HASH, BINRY
	A table used while HASHing an identifier. HASH stores up to five characters at a time in TH which is superimposed over T6, T7, & T8. These are then added together and the sum placed in T9 in the following manner. Starting with the left most, up to five characters are stored at a time in TH. These are right justified with leading blanks. When completed, they are taken from TH, ten bits at a time (truncating if any result exceeds ten bits), added together in T9. T9 plus one will then be placed in T4. T4 is then used as an index to DISH.			
TREGN	Subscripted item from table DICT	Express	Not used	
	If CIAS equals procedure: Specifies the starting level of internal temporary storage required by the procedure.			
TREGNM	Subscripted item from table DICTM	Express	Not used	
	Identical to TREGN in the dictionary.			
TSTD	Item	Express	Used by:	MP
	STC variable used to save one card of direct code before it is output.			
TSTIM	Procedure			
	Procedure which reads the Hollerith input tape.			
TSTR	Item	Express	Used by:	MP, TSTIM
	STC variable into which one card at a time is read from the Hollerith input tape (HIN).			
TXKT	Item	Express	Used by:	LOOK, CHXTX, Main program
	Set to the first byte in either the first or second half of IDNT. If the first half of IDNT is being used by LOOK to store a constant or identifier, TXKT is set to the second half and vice versa.			

<b>TY</b>	Subscripted item	Local	Used by:	LOOK
	Item in unnamed table that contains type value for each special character listed in item SC.			
<b>TYP</b>	Item	Local	Used by:	CAN
	Item synonymous with TYPV in DICT.			
<b>TYP</b>	Subscripted item	Local	Used by:	PDICT
	STC variable in an unnamed table containing preset data, which gives a mnemonic name for each possible value of the DICT item TYPV.			
<b>TYPV</b>	Subscripted item from table DICT	Express	Used by:	CAN, IMP, PDICT, IDEAL, POOL
	Specifies the type of constant, item, subscripted item or array:			
	0 = null	7 = status (S)		
	1 = fixed point (A)	8 = octal (O)		
	2 = Boolean (B)	9 = STC (T)		
	3 = dual fixed point (D)	10 = dual inter (K)		
	4 = floating point (F)	11 = dual octal (Q)		
	5 = Hollerith (H)	12 = (error)		
	6 = integer (I)			
	<u>Note:</u> For FILE, set to 7.			
<b>TYPVM</b>	Subscripted item from table DICTM	Express	Used by:	IMP, IDEAL, CAN
	Parallel to TYPV in the dictionary.			

10 July 1962

59

TM-555/020/00

URE1      Subscripted item from table PURE      Express      Used by:    MP  
            Contains first character byte in IDNS of undeclared procedure name.

URE2      Subscripted item from table PURE      Express      Used by:    MP  
            Contains number of characters in undeclared procedure name.

VAL	Item	Local	Used by: CAN
	This contains the value, actual number, of an A or E factor.		
VALK	Subscripted item from table SETK	Express	Used by: MP
	Contains preset data for setting the value of K1, the left term in a triplet.		
VALU	Item	Express	Used by: SCON
	VALU is a string containing status values. The stati are stored with a separator. Thus, V(ABC) will be stored as ABCΔ where Δ is an octal 77(STC) or, effectively, a blank.		
VCATC	Item	Express	Used by: MP
	Contains CATC for CAT entry following the TERM operator.		
VCATF	Item	Express	Used by: MP
	Contains CATF for CAT entry following the TERM operator.		
VLUE	Subscripted item from table CED	Express	Used by: MP
	Contains preset data which is the K1 values which immediately precede a unary operator.		
VP	Item	Local	Used by: LOOK
	Saves subscript value when processing a floating point constant.		
VPOS	Subscripted item from table STAT	Express	Used by: SCON
	This is the first byte of status values for this particular entry.		

10 July 1962

61

TM-555/020/00

WOHIN      Subscripted item from table BCE      Express      Used by:    MP  
Item in table BCE which gives entrance value for switch S3 for  
each separator, operator, or bracket.

WORD      Item      Express      Used by:    MP, FNRC  
Stores identifier being processed while it is being compared  
against item DELIM to determine whether it is a recognizable part of  
speech.

10 July 1962

62

TN-5 55/02/00

XX	Item	Express	Used by:	MP
	Temporary storage variable.			
XX	Input Parameter	Local	Used by:	LOOK
	Input parameter giving reason for procedure call.			

10 July 1962

63

TM-555/020/00

YY	Output Parameter	Local	Used by:	LOOK
	Output parameter giving type of part of speech.			
YY	Input Parameter	Local	Used by:	ERROR
	Input parameter to ERROR which contains the error type to be printed.			

10 July 1962

64

TM-555/020/00

ZZ            Output Parameter                            Local        Used by:    LOOK

Output parameter giving number of characters.

ZZ            Item                                        Express    Used by:    Main program

Set to number of entries for output of final segment of CAT.



10 July 1962

65

TM-555/020/00

C. DESCRIPTION OF THE GEN1 MAIN PROGRAM (MP) REGIONS

1. a. Name - JG~~GIN~~  
b. Description - Does initialization, checks for START at the beginning of the JOVIAL source program, and looks at the next part of speech.  
c. Procedures called - TSTIM, LOOK, ERROR  
d. Operations -
  1. Sets machine dependent variables CDCL1, and CLSIN.
  2. Presents simple variables and tables FIND and CAT to zero.
  3. Goes to procedure IDEAL for its initialization pass.
  4. Reads in the first card and checks for the end of a set of jobs.
  5. Puts START in CAT and checks for a START operator as first part of speech. If no START, prints error message.
2. a. Name - ST~~UFF~~  
b. Description - Finds proper entry in the legality matrix and processes the middle term of the current triplet accordingly.  
c. Procedures called - LOOK  
d. Operations -
  1. Looks at the next part of speech.
  2. If part of speech is "DIRECT", goes to process direct code statement.
  3. If part of speech is "\$" and is inside of DIRECT-JOVIAL brackets, goes to finish processing ASSIGN statement.
  4. Looks at the next part of speech to set the value of the right term in the triplet.
  5. Checks for DIRECT as the next operator, or A( (accumulator designator) as the current part of speech.
  6. Finds the entry in the legality matrix corresponding to the left and right terms in the triplet and goes to switch S1.

3. a. Name - CHEND  
b. Description - Checks for TERM being bypassed in case of an error.  
c. Procedures called - none.  
d. Operations -
  1. Checks for the TERM operator being missed because of a call to FNRC.
  2. In case TERM has been missed, the TERM operator is put in the CAT table and the program is completed normally.
4. a. Name - X100  
b. Description - Prints CAT table entry if desired and increments index in CAT.  
c. Procedures called - RECRD.  
d. Operations - Calls RECRD to print the last CAT entry and increments the index in CAT (11).
5. a. Name - X2  
b. Description - Processes separators, operators and brackets.  
c. Procedures called - ERROR, FNRC.  
d. Operations -
  1. Checks for part of speech with type less than 18 (special character).
  2. If not, checks identifier against table of operators, and if found, sets CATC and CATF, and sets K1, the value of the left term in the next triplet.
6. a. Name - X14  
b. Description - Checks legality matrix entries of 6, 7 or 8.  
c. Procedures called - ERROR, FNRC.  
d. Operations -
  1. Checks the variable MTYP to see which legality matrix entry was encountered and goes to the spot appropriate for an entry of 2, 3 or 4.

## 7. a. Name - X13

b. Description - Makes special checks after processing the middle term of a triplet.

c. Procedures called - RECRD.

d. Operations -

1. Checks to see if a procedure is being processed and if so goes to do special checks.
2. Prints last entry in CAT if desired and increments the index in CAT by 1.
3. Checks to see if the next part of speech to be processed is DIRECT, if IDEAL has been executed, or if an ASSIGN statement has just been completed.
4. Sets the value of middle term of the next triplet to be processed.

## 8. a. Name - X10

b. Description - Makes special checks on parts of speech with type less than 18.

c. Procedures called - ERROR, FNRC.

d. Operations -

1. Sets K1 value for the left term of the next triplet to be processed.
2. Checks for period after a statement label. If so, and the statement label is not a procedure output parameter, the label is saved for error printout.
3. Checks for a right parenthesis. If so, checks to see if it is being used in a nested modifier to keep track of which parenthesis should be eliminated from CAT. Checks Boolean item RTPAR to see if this parenthesis should be eliminated.
4. Sets CATC and CATF for this part of speech.
5. Checks for dollar sign. If so, checks for the end of an item switch to subtract one from the number of switch points. If a switch is being processed (SWEP) set switch dollar sign indicator (SWDS) to on and SWEP to off. If the nested modifier parenthesis table is not cleared out, print error message and clear out table. Increment the statement counter, and check to see if the capacity of CAT has been reached. If so, write out CAT.

8. d. 6. Checks for plus (+) or minus (-). If so, checks for a unary operator. A unary plus is eliminated from CAT, and the value of a unary minus is changed.
  7. Checks for left parenthesis. If so, and a nested modifier is being processed, the appropriate counter is incremented.
  8. Checks for a switch being processed (SWEP on). If so, counts commas for the number of switch points, and counts parentheses to determine whether an identifier inside of a switch is a statement label or not.
  9. Goes to process next triplet.
9. a. Name - X15
  - b. Description - Makes special checks when procedure is being processed.
  - c. Procedures called - none
  - d. Operations -
    1. Sets PTYP to proper setting for input or output parameter.
    2. Checks for BEGIN and END and counts brackets accordingly. When the bracket counter reaches zero, the CAT entries for the procedure are written out.
    3. When procedure is completed, if JOVIAL library is being processed, return is made to read more library. If not, goes back to process next triplet.
10. a. Name - X3
  - b. Description - Processes constants, simple variables, files, and single letter subscripts declarators.
  - c. Procedures called - ERROR, FNRC, ENTER, LOOK.
  - d. Operations -
    1. Checks for a constant. If so, a call to ENTER is made.
    2. Checks for a single letter subscript. If so, an entry is made in CAT.
    3. In all other cases, a simple variable is assumed and a call is made to ENTER. Upon return, a check is made for a procedure, switch or close declaration, or a table, item, etc., declaration.

11. a. Name - X26
  - b. Description - Processes the first part of a switch declaration.
  - c. Procedures called - ERROR, LOOK, ENTER, RECRD
  - d. Operations -
    1. Turns on SWEP and initializes parenthesis counter BKCT.
    2. Makes a call to ENTER with the switch name.
    3. Checks to see if switch is an item or numeric switch. If switch is an item switch, NORI is turned on and a call is made to ENTER with the item name.
    4. Puts entries in CAT for the parts of speech in the declaration processed so far.
12. a. Name - X27
  - b. Description - Processes beginning of procedure declaration.
  - c. Procedures called - ERROR, LOOK, RECRD, ENTER
  - d. Operations -
    1. Checks for PREP being already on. If so, reports error and writes out previous portion of CAT as a procedure.
    2. Turns on PREP (procedure indicator) and sets bracket counter (BRCTR) to zero.
    3. Calls ENTER with procedure name.
13. a. Name - X28
  - b. Description - Processes beginning of close declaration.
  - c. Procedures called - RECRD, ERROR, FNRC, LOOK, ENTER
  - d. Operations -
    1. Calls ENTER with close name.

14. a. Name - X4
- b. Description - Processes modifier or procedure name.
- c. Procedures called - ERROR, FNRC, ENTER.
- d. Operations -
1. Compares identifier with list of modifiers in table BCE and if successful, an entry is made in CAT.
  2. If not, a call is made to ENTER with the procedure name.
15. a. Name - X30
- b. Description - Processes the table name modified by NENT, ENT, ENTRY, NWDSN, ALL.
- c. Procedures called - LOOK, ENTER.
- d. Operations -
1. Calls ENTER with the table name modified.
  2. Adds one to index in table FIND, and adds one to entry in FIND to indicate a new modifier has been encountered.
16. a. Name - X31
- b. Description - Makes entry in CAT for modifier ABS.
- c. Procedures called - none.
- d. Operations -
1. Makes entry in CAT for modifier ABS.
17. a. Name - X33
- b. Description - Does additional processing on MANT, CHAR, ODD, POS.
- c. Procedures called - ERROR, LOOK.
- d. Operations -
1. Adds one to the index in table FIND and adds one to entry in FIND to indicate a new modifier has been encountered.
  2. Subtracts one from index in CAT to allow for modifier going in CATB.

18. a. Name - X5
- b. Description - Processes BIT, BYTE, HEAD or subscripted variable.
- c. Procedures called - ERROR, FNRC, ENTER.
- d. Operations -
1. Checks the identifier against BIT, BYTE or HEAD, and if a match is made, makes an entry in the CAT table.
  2. If no match is made, a call to ENTER is made with the subscripted variable name.
19. a. Name - X6
- b. Description - Processes statement labels and close labels.
- c. Procedures called - ERROR, FNRC, ENTER.
- d. Operations -
1. Checks for label within a procedure heading. If so, sets PSSL to 1.
  2. Calls ENTER with proper setting of PARAM for statement or close label.
20. a. Name - X7
- b. Description - Decides what to do for matrix value of 6, 7, or 8.
- c. Procedures called - none.
- d. Operations -
1. Checks for constant or single letter subscript and if so, goes to process them.
  2. Sets MTYP according to legality matrix value and goes to check for separator, operator or bracket.



- 21. a. Name - DRCT
  - b. Description - Processes direct code.
  - c. Procedures called - TSTIM.
  - d. Operations -
    - 1. Presets indicators and reads in one card of direct code and moves it to TSTD.
    - 2. Searches for word JOVIAL. If found, sets indicator (INDIR) to say no more direct code and puts entry in CAT for last batch of direct code and then returns to processing in the regular manner.
    - 3. Searches for word ASSIGN. If found, puts entry in CAT for last batch of direct code and returns to process the ASSIGN statement.
    - 4. If neither JOVIAL nor ASSIGN is found on the next card, the card is output to the direct code file (DIRCT), and another card is looked at.
- 22. a. Name - PRAC
  - b. Description - Processes accumulator designator in ASSIGN statement.
  - c. Procedures called \_ RECRD, LOOK, ERROR, DTOB.
  - d. Operations -
    - 1. Places proper entry in CAT for accumulator designator in ASSIGN statement.
- 23. a. Name - HTDS
  - b. Description - Processes dollar sign ending ASSIGN statement.
  - c. Procedures called \_ RECRD.
  - d. Operations -
    - 1. Sets entry in CAT for the dollar sign and goes back to process more direct code.

24. a. Name - PRIP
- b. Description - Takes care of error when procedure declaration appears within procedure declaration.
- c. Procedures called - ERROR.
- d. Operations -
1. Prints error message and goes to write out previous procedure if necessary.
25. a. Name - X12
- b. Description - Finishes processing after TERM operator has been found and writes out last portion of CAT.
- c. Procedures called - CHXT, ENTER, RECD.
- d. Operations -
1. Writes out the portion of CAT remaining in memory.
  2. Checks to see if a statement label follows TERM. If so, calls ENTER to process label.
  3. Makes an entry in CAT for the dollar sign following TERM.
  4. Checks to see if a library tape is available. If not, goes to finish up processing.
  5. Searches dictionary (DICT) for undeclared procedures. If none, calls ENTER to allow IDEAL to do its final processing and writes out portion of CAT remaining in memory.
  6. Returns to control program.
26. a. Name - IO
- b. Description - Processes I/O operators.
- c. Procedures called - LOOK
- d. Operations -
1. Sees the proper entry is made in CAT for INPUT or OUTPUT following OPEN or SHUT operator.

10 July 1962

75  
(page 76 blank)

TM-555/020/00

27. a. Name - GTPR

b. Description - Matches the library index against the list of undeclared procedure names, to find additional procedures in the library that must be called in from the library file.

c. Procedures called - TSTIM, LOOK, ERROR, IDEAL.

d. Operations -

1. Presets simple items and saves CAT entry following the entry for TERM.

2. Zeros out CAT and reads first card in from the library file.

3. Compares the list of undeclared procedure names with the procedure names in the library index. When a match is found, any additional procedures listed as being used which are not already in DICT are entered in DICT via a call to IDEAL.

28. a. Name - GTP5

b. Description - Compares undeclared procedure names against procedure names as they are encountered in the procedure declarations in the library file.

c. Procedures called - LOOK, CHTXT, DTOB, TSTIM.

d. Operations -

1. Compares the procedure name from the declaration in the library file with each name in the list of undeclared procedures. If a match occurs, the declaration is processed as though it were part of the JOVIAL source program.

2. If no match is found, the cards in the procedure declaration are skipped, and the next procedure name is examined.

29. a. Name - GTP7

b. Description - Place TERM entry in CAT and goes to finish processing and output CAT.

c. Procedures called - RECRD

d. Operations -

1. Places TERM entry in CAT.

2. Checks for statement label after TERM, and if so, places entry in CAT, adds CAT entry for "\$", and goes to finish processing and write out CAT.

10 July 1962

77

TM-555/020/00

D. DESCRIPTION OF THE GEN1 PROCEDURES

BINARY

- a. Unsigned integers of 5 or less digits, coded in 6-bit STC, are converted to binary integers. A +0 is ensured against the possibility of a hardware distinction between +0 and -0. If a non-digit is detected, +0 results. Output is available for immediate assignment.
- b. Input parameter  
    NUMB = number of digits in the integer for conversion.  
  
    Output parameter  
    BINARY = converted results ready for immediate assignment.
- c. Local variables  
    T1  
    T10
- d. Express variables  
    TH  
    T8  
    TEXT  
    IDNT
- e. No procedures called.
- f. Entered from  
    IMP  
    IDEAL
- g. A powers of 10 multiplier is started at  $10^0$ , i.e., one. Digits are processed from the right to the left. Each digit is examined for the range 0 to 9, i.e., octal 60 to octal 71, in 6-bit STC. Failure results in a +0 output and immediate exit. Each digit is converted by subtracting octal 60, multiplying by the powers of ten multiplier, and accumulating the sums of the products. The powers of ten multiplier is multiplied by ten prior to processing the next digit left. The integer is accessed in IDNT using TEXT and NUMB. Results are prepared for immediate assignment (accumulator stored).

BTOD

- a. BTOD is a function converting a binary integer to STC.
- b. Input parameter  
    AA = number to be converted.  
    No output parameters.
- c. No local variables.
- d. No express variables.
- e. No procedures called.
- f. Entered from  
    CAN  
    FDICT  
    RECRD  
    SCON
- g. The binary integer AA is converted to an STC integer. Results are prepared for immediate assignment.

CAN

- a. CAN analyzes constants and exits with the constant converted to conform with its TYPV or the TYPV of the item to which it is associated.

- b. No input parameters.

No output parameters.

- c. Local variables

BF	IM	R3
BRT	L1	R4
DAT	L2	SDT
DIF	L3	SEP
DOT	NEG	SGN
DUL	OCT	STC
DUN	R1	TO
ERR	R2	TYP
		VAL

- d. Express variables

BRGTM	TEXT
IDNT	TYPV
NCHAR	TYPVM
PDATM	T2
PDAT	T5

- e. Procedures called

BTOD  
ERROR

- f. Entered from

IDEAL  
IMP  
SCON

- g. The constant analysis routine, CAN, is called to process constants. This procedure analyzes and, if necessary, alters inputs for processing. CAN works with the contents of IDNT, indexed by TEXT. In case of an error, the contents of IDNT are not to be destroyed, thus a buffer string, IM, is used to hold the 128 characters for CAN. CAN actually scans IDNT one character at a time in building IM; when completed, the entire entry is placed back in IDNT. In processing, decimal points are removed from the

constant. A buffer, BF, is used for transferring a portion of the constant and then overlaying the period while keeping track of its position for the E factor.

## CAN

<u>INPUT FROM</u> <u>IDNS</u>	<u>OUTPUT FORM IN IDNS</u>	
	<u>DATA</u>	<u>NON-DATA</u>
1.03	103E-02	103E-02
10.3E-1 A5	103E-02	103E-02+05
103.E-2	103E-02	103E-02
103E1	TYPV=2,6,10: 1030	1030
	TYPV=1,3,4: 1030E+00	
	TYPV=2,6,10: 103	103
	TYPV=1,3,4: 103E+00	

If the constant is not an A or F TYPV, output depends on TYPV. If TYPV=4 or PDAT=1, unsigned constants are prefixed with a plus. Boolean constants greater than one will be flagged as error. If the constant has a +E factor and no decimal point, CAN generates zeros. A-E factor with no decimal point is considered an error. If data described as being unsigned but is signed, the "plus" will be removed, if "minus" it will be removed and also flagged as an error. If a constant is described as being signed but no sign is included, a + will be generated, never a -.



CHTEXT

a. CHTXT reverses the index in IDNT.

b. No input parameters.

No output parameters.

c. No local variables.

d. Express variables.

TEXT, TOUT

e. No procedures called.

f. Entered from

Main program

g. CHTXT exchanges the values of TEXT and TOUT.

DTOB

a. DTOB is a function converting an STC integer to binary.

b. Input parameter  
AA = number to be converted.

No output parameter

c. No local variables.

d. No express variables.

e. No procedures called.

f. Entered from

DMP, Main Program

g. The STC integer AA is converted to binary integer. Results are prepared for immediate assignment

ENTER

a. ENTER calls IDEAL.

b. No input parameters.

No output parameters.

c. No local variables.

d. Express variables

SWBP	NCHAR
NORI	CATC
KI	CATF
HKCT	II
PARAM	EXCID

e. Procedures called

IDEAL

f. Entered from

POOL, Main Program

g. ENTER contains a call to IDEAL. A check is made in ENTER to make certain that the input to IDEAL is correct for labels inside SWITCH declarations.

ERROR

a. ERROR prints the error messages.

b. Input parameter

YY = error type to be printed.

No output parameters

c. No local variables.

d. Express variables

ERTY	NCHB	STMT
ERT	IDNS	STCTR
PRBP	PRST	TEXT
PRCHD	PRSC	IDNT
FCHB		

e. No procedures called.

f. Entered from

Main Program  
IMP  
LOOK  
SCON  
TSTIM  
CAN  
POOL  
IDEAL

g. ERROR prints error messages giving type of error and location (main program or procedure name and statement label + number of statements). The input parameter to the procedure gives type of error.

FNRC

a. FNRC provides for restarting operation after an illegal triplet has been detected.

b. No input parameters

No output parameters

c. No local variables

d. Express variables

TYPE	KVAL
LL	WORD
NCHAR	IDNT
NXTCR	DELIM
K1	EXTRA

e. Procedures called

LOOK

f. Entered from

Main Program

g. If an error is found in processing a triplet, the scheme is disrupted and it is necessary to set the value of K1, the left term of the next triplet to be processed. FNRC searches the source program until it comes upon a recognizable part of speech. This part of speech is then used as the left one in a triplet and processing continues from this point.

GET2

a. The procedure LOOK is called to bring the next part of speech into IDNT. Access references to IDNT are bookkept and a Boolean indicator is set indicating that LOOK has been called.

b. No input parameters.

No output parameters.

c. Local variable

CHR

d. Express variables

IDNT	TEXT
IDOL	TIPE
L1	T1
NCHAR	T5
NXTCR	

e. Procedure called

LOOK

f. Entered from

IDEAL  
IMP  
SCON

g. 1. Set TIPE = L1 and NCHAR = NXTCR.

2. Call the procedure LOOK with calling parameters:  
2 = read next part of speech,  
L1 = part of speech code,  
NXTCR = number of characters, if supplied.

3. Test TIPE for code 13 (-) or 14 (+). If 13 or 14, call LOOK again and prefix correct sign.

4. If NCHAR = 0, set NCHAR = 1.

5. Set:

T1 = TIPE,  
T5 = NCHAR,  
IDOL = 1 (true).

GET3

a. The procedure LOOK is called, repeatedly, if necessary, to bring the first two parts of speech following the next "\$" separator into IDNT. Access references to IDNT are bookkept, and an indication that LOOK has been called is set.

b. No input parameters.

No output parameters.

c. No local variables.

d. Express variables

IDOL  
L1  
NCHAR  
NXTCR  
TIPE  
T1  
T5

e. Procedure called

LOOK

f. Entered from

IDEAL  
IMP

g. 1. Set T1 = TIPE, TIPE = L1, and NCHAR = NXTCR.

2. Call the procedure LOOK with calling parameters of:

2 = read next part of speech,  
L1 = part of speech code,  
NXTCR = number of characters, if supplied.

3. Test T1 for code of 4(\$). If T1 is not 4, repeat the process.

4. If NCHAR = 0, set NCHAR = 1.

5. Set:

T1 = TIPE  
T5 = NCHAR  
IDOL = 1 (true)

HASH

a. JOVIAL identifier (i.e., labels and constants) are converted from 6-bit STC inputs to an index value between 1 and 1024 of a rapid access dictionary search table, DISH.

b. No input parameters.

No output parameters.

c. Local variable

T13

d. Express variable

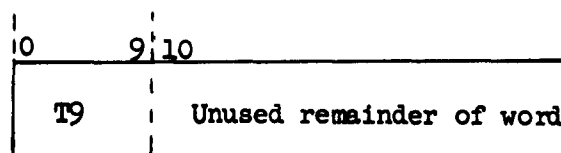
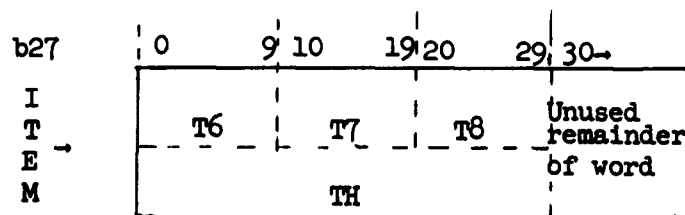
FOND	TEXT	T6
IDNT	TH	T7
NCHAR	T3	T8
NEXT	T4	T9

e. No procedures called.

f. Entered from

IDEAL  
POOL

g. A schematic of table TPY follows





Identifiers in 6-bit STC are transferred from IDNT to TH starting with the leftmost character of IDNT. Five characters are transferred at a time. Fewer than 5 characters may be transferred if the number of characters in the identifier is not divisible by 5. In this event, the odd number of characters are right justified in TH with STC blanks assigned to the left character positions. After each transfer, the 10-bits in T6, T7 and T8 are added to and stored in the accumulated sum, T9. Sums exceeding 1023 are truncated by the assignment to T9 resulting in a sum modulo 1024. The process is repeated until all characters in the identifier have been "hashed." The following assignments then occur before exit is made:

$$T4 = T9 + 1$$
$$T3 = \text{NEXT (next available byte number in IDNS)}$$
$$\text{FOND} = 0 \text{ (false)}$$

IDEAL

- a. IDEAL processes declarations, totally or partially, invoking the aid of specific processing procedures where necessary. It prepares the dictionary (DICT) entries defining labels and constants, processes preset parameter constants included with declarations, builds IDNS, VALU, and the tables STAT, OVER, and DISH.

- b. Input parameters

RSON = reason for entry; identical to PARAM

NUMB = number of characters in the identifier or constant

Output parameters

CLSS = CAT table class of processed variable

CHNL = channel in the dictionary defining the identifier or constant

- c. Local variables

EXES  
T10  
T16  
T17

- d. Express variables

BRGT	FBIT	NCHM	PDAT	TEXT
CHL1	FCHB	NDICT	PDATM	TFRM
CHL1M	FCHM	NEXT	PREP	TFRMM
CHL2	FIRST	NOCN	PTYP	TIPE
CLAS	FOND	NOVER	SCOPE	TYPV
CLASM	FPNM	NSTAT	SIND	TYPVM
CMPO	FPNMM	NVAL	SINDM	T1
DEFN	IDNS	NVALU	SIZE	T2
DEFNM	IDNT	NXTCR	SIZEM	T3
DELIM	IDOL	OLAY	SPEC	T4
DENT	INUS	OVEN	STAT	T5
DICT	LAST	OVER	STC	T11
DICTM	L1	OVID	SUM	T12
DIMN	MAXN	OVLI	SWBP	T13
DIMZ	NARDA	PACK	SWCN	T14
DISH	NCHAR	PACKM	TBBP	T15
DMAIN	NCHB	PCHN	TCHN	VLUE

## e. Procedures called

BINARY	IMP
CAN	PDICT
ERROR	POOL
GET2	SCON
GET3	SRCH
HASH	

## f. Entered from

ENTER

- g. IDEAL begins processing by testing the identifier in IDNT with the switch DECOR. If the identifier is CLOSE, SWITCH, or PROC, control is returned to the main program with CLAS and CHNL set to indicate the JOVIAL primitive encountered. Switch and procedure declarations are partially processed by IDEAL. If the identifier is one of the following, the declaration is completely processed:

ITEM (both subscripted and non-subscripted, including preset parameter constants)

TABLE

MODE

FILE

OVERLAY

STRING

ARRAY (including preset parameter constants for ARRAYS of three or less dimensions)

Parameter items preset to, and defining a non-subscripted item, are also processed. Variable definitions are entered in the dictionary and the identifier and preset parameters are entered in IDNS.

In processing, HASH and SRCH are called to see if an entry having the same name has previously been entered in the dictionary. If so, the present item is tested to see if it agrees in usage with that already in the dictionary. Primary legality checks are accomplished with table LEGAL. If two entries have the same name but different usage, a new entry is made in DICT. When IDEAL encounters a variable that is used without having been defined previous to its use, and if there is a compool, IDEAL calls procedure POOL to extract its definition from the compool and enter it in DICT. If there is no compool, or if the variable is not defined therein, it is entered into the dictionary according to the prevailing MODE declaration.

10 July 1962

93

TM-555/020/00

IDEAL also calls IMP to process simple and subscripted items and strings;  
CAN to analyze constants; GET2 to get the next part of speech from LOOK;  
GET3 to look for the next \$; BINRY to convert an STC integer to binary;  
SCON to process status variables.

IMP

- a. IMP processes that portion of declarations called the item description including the preset parameter constant if it is a single constant in a non-subscripted variable declaration. Processed ITEM descriptions are stored in channel 3 of DICTM and associated parameters, if present, are stored in channel 4 of DICTM.

- b. No input parameters.

No output parameters.

- c. Local variables

STC

- d. Express variables

BRGTM	IDNS	NVAL	SIZEM	TYPVM
CHL1M	IDNT	PACKM	SPEC	T1
CHL2M	L1	PDAT	TBBP	T2
CLASM	NCHAR	PDATM	TCHN	T3
DIMMM	NCHBM	RNGIM	TEXT	T5
FCHEM	NDICT	RORTM	TIPE	T12
FPNMM	NEXT	SINDM	TYPV	T13

- e. Procedures called

BINRY  
CAN  
DTOB  
ERROR  
GET2  
GET3  
SCON

- f. Entered from

IDEAL

- g. The procedure IMP processes that portion of a declaration which is called "item description," plus any other information following the item description and preceding the terminating "\$" of a given variable declaration. It processes, therefore:

1. Non-subscripted ITEM declarations following the identifying name to the terminating "\$" separator.

2. ARRAY declarations following the last dimension to the terminating "\$" separator.
3. STRING and subscripted ITEM declarations following the identifying name to the terminating "\$" separator.
4. MODE declarations following the primitive "MODE" to the terminating "\$" separator.

Variables with type-code of "A" (fixed-point arithmetic) are given type-code of "A" if bits right appears in the ITEM description (zero bits right included, if specified). Otherwise, type-code of "I" is given. If type-code of "I" is declared (not in language) this is also processed as an integer. Type-code "D" is similarly processed into "D" (dual fixed-point) and "K" (dual integer). Type-code of "S" (status) has SIZE (number of bits) computed in absence of a declared number of bits. Non-subscripted ITEM declarations of the form

ITEM ITEM'NAME constant \$

are not processed by IMP. Information gleaned by IMP is stored in channel 4 (index 3) of DICTM. Pre-assigned parameter constants associated with a non-subscripted variable are processed into channel 5 of DICTM.

LOOK

a. Finds the next part of speech in the JOVIAL source program. An output parameter is set to the number designating type, and identifiers and constants are placed in IDNT.

b. Input parameter

XX = reason for entry (always set to 2, 0 and 1 are not used)

Output parameter

YY = type (see values at end of procedure description)

ZZ = number of characters in identifier or constant

c. Local variables

CB	INC	SB
CDIT	LB	SC
CHCT	LCHAR	SV
DB	OSB	TCH
FCHAR	PB	TY
HB	PEB	VP
HL	PP	

d. Express variables

CARD	IDNT	NC	NX
CATC	IL	ND	PREP
CATF	JJ	NDEFS	PRSC
CDI	J1	NDICT	SAV
CHAR	KD	NEXT	STCTR
DERBP	KDA	NI	TCD1
DFB	KDB	NO	TEXT
DI	KDC	NOVER	TXXT
FB	L1	NU	

e. Procedures called

DTOB  
ERROR  
TSTIM

f. Entered from

Main program  
FNRC  
GET2  
GET3

- g. The procedure looks at one character at a time, from item CARD, until a part of speech has been terminated. If a comment bracket is encountered (''), the entire comment is bypassed, the ending symbol being either another comment bracket or a dollar sign (\$). If the identifier DEFINE is encountered, the declaration following is processed by LOOK, and the information stored in item KD and the appropriate entries made in the define table (contains items KDA, KDB and KDC). In both the above cases, the next part of speech is obtained by LOOK as the output from the procedure.

LOOK recognizes basically three kinds of characters: (1) special characters, (2) numbers, and (3) letters.

- (1) When a special character is recognized the following checks are made:
- a. If the number of characters already processed for this part of speech is equal to zero, the character is compared with the single characters in item SC. If a match is found, appropriate action is taken. If the character can be the first of a 2 character pair, a 2 character set is compared against those in item SC. If the special character is not found to be one recognized by JOVIAL, and it is not the ending character of defined information, an error message is printed and the next part of speech is gotten.
  - b. If the special character is a prime (') with no prime following it, it is assumed to be part of an identifier and the next character is looked at.
  - c. If a constant is being processed (CB set to 1), the character is checked to see if it is part of the constant. If so, it is handled accordingly. If not, the constant is terminated and exit is made from the procedure.
  - d. If the character is a left parenthesis (()) and it is the second character processed, the two characters are checked against item SC for possible special meanings. If a match is found, switch GT is used to continue processing. If no match is found, the part of speech is terminated.
  - e. If none of the above steps are taken, the special character terminates a previous part of speech and the output parameters are set.

If the part of speech just finished is an identifier the following additional decisions must be made:



1. If the identifier is the label in a `DEFINE` declaration, the rest of the declaration is processed, and then the next part of speech is obtained for return from the procedure.
  2. If the number of defined identifiers is greater than 0, the current one must be compared against the existing list, and if a match is obtained, the definition must be processed as though it were part of the JOVIAL source program.
  3. The identifier must be checked to see if it is either `"DEFINE"` or `"DIRECT"`. In the former case, the declaration is processed. In the latter, the output parameter type must be modified.
- f. If the part of speech just processed is a dollar sign (\$), the following decisions must be made.
1. Checks subscripted item `BRCN` to see if an uneven number of brackets was encountered in the statement terminated by \$. If so, error 23 is reported.
  2. If a procedure is being processed, the statement counter for procedures is incremented by one. If not, the statement counter for main program is incremented by one.
  3. `DICT`, `OVER` and `IDNS` are checked to see if they have overflowed during the previous statement. If so, an error message is printed and the processing is terminated. If not, the position in `IDNT` is reversed, and return is made from `LOOK`.
- (2) When a number is recognized, if it is the beginning of a new part of speech, the constant indicator is turned on (`CB`). The character is put into `IDNT`, the character count is incremented by one (`CHCT`) and the next character is looked at.
- (3) When a letter is recognized, indicators are set, the character is put into `IDNT`, the character count is incremented by one (`CHCT`) and the next character is looked at.

In general, identifiers and constants are placed in `IDNT` to be available to other parts of the generator. Each call to `LOOK` causes the index in `IDNT`, `TEXT`, to be changed with `TXXT`, to alternate halves of `IDNT`. Upon return from `LOOK`, `TEXT` indicates the half of `IDNT` to be used next time through the procedure.

Everytime the index associated with item `CARD` is incremented (`J1`), a call to `TSTIM` is made to insure that a card will be read in when necessary.

10 July 1962

99

TM-555/020/00

The output from LOOK giving type of part of speech is as follows:

1 = .	13 = -
2 = ,	14 = +
3 = =	15 = *
4 = \$	16 = /
5 = ==	17 = **
6 = ...	18 = '
7 = (	19 = ''
8 = )	20 = constant
9 = (\$	21 = identifier
10 = \$)	22 = subscript
11 = (/	23 = DIRECT
12 = /)	24 = A(

MUV

a. Places one character in item IDNT.

b. No input parameters.

No output parameters.

c. Local variables

AA  
COUNT

d. Express variables

CHAR  
FBX  
IDA  
IDB  
IDC  
IDD  
IDE  
IDF  
IDG  
IDH  
TEXT

e. No procedures called.

f. Entered from

LOOK

- g. 1. The first time the procedure is entered in one execution of procedure LOOK, (FB set to one), the entrance value to switch SW is initialized, and the half of IDNT to be used is determined from TEXT.
2. Each time the procedure is called, one character from item CHAR is stored in IDNT. Consecutive bytes of IDNT are used by storing in a different subscripted item each time the procedure is called.

PDICT

- a. Under central control, edit and store on tape in off-line print format the identifier storage item, IDNS, and the dictionary, DICT.

- b. Input parameter

FIRST = an index value indicating the final (high order) channel of DICT to be edited

Output parameter

LAST = an index value indicating the final (high order) channel of DICT to be edited.

- c. Local variables

AA	TITLE
CLASS	TYP
DEF	TFR
PAC	

- d. Express variables

CLAS	PDAT	BRGT
DEFN	PACK	FPNM
TYPV	SIND	PDCTY
FCHB	OLAY	TAPE
NCHB	TFRM	NEXT
CHL1	SIZE	IDNS
CHL2	FBIT	DEBUG

- e. Procedures called

BTOD

- f. Entered from

IDEAI

- g. 1. If PDCTY is not true, exit.

2. Edit IDNS to print 100 characters per line with blanks in position 1 and positions 101 to 120, using item TAPE as a 120 character buffer. Convert tape from STC to hardware Hollerith by calls to the procedure CONV.

3. Output each line to be printed from the item TAPE through the file DEBUG.
4. Output enough lines to completely include IDNS.
5. Output a DICT heading preset in the item TITLE through the file DEBUG.
6. Edit 1 line of DICT for each channel to be printed.
7. Step counter AA starting with FIRST and repeat process for each channel in DICT until AA exceeds LAST.
8. Edit each line using the buffer item TAPE. Convert TAPE to hardware Hollerith with calls to the procedure CONV.
9. Output each DICT channel as edited in TAPE through file DEBUG.

POOL

- a. POOL searches the compool for definitions of identifiers used by the program that are not previously defined. POOL is presented with an identifier and an indication of the type of variable whose definition may be extracted from the compool. If a definition is found it is entered in the dictionary (DICT) and associated tables. In addition to setting items in the reserved dictionary entry, POOL will, under certain circumstances, add entirely new entries to the dictionary.

- b. No input parameters.

No output parameters.

- c. Local variables are compool dependent.

- d. Express variables

BRGT	DIMN	INUS	NVALU	STC
CHL1	FBIT	NARDA	PACK	TEXT
CHL2	FCHB	NCHAR	PARAM	TFRM
CLAS	POND	NCHB	PCHN	TYPV
DEFN	FPNM	NDICT	PDAT	T3
DENT	IDNS	NEXT	PRBR	T4
DICT	IDNT	NSTAT	SIZE	T13

- e. Procedures called

ENTER  
ERROR  
HASH  
SRCH

- f. Entered from

IDEAL

- g. The procedure POOL is compool dependent. There is a different POOL for each compool, and thus, for each compiler. However, all POOL procedures will operate in the same manner.

If the express Boolean item CMPOL is set to true, POOL will be entered. The identifier in IDNT accessed by TEST and NCHAR, and identified as to type by PARAM, will be looked up in the compool and its definition entered in the reserved dictionary channel specified by the value of T13. DEFN of channel T13, is set to 2 to indicate a compool-defined identifier. An ITEM, TABLE, STRING, ARRAY, LIKE TABLE, PROCEDURE, or a program may be defined in this manner. Parameter items may not be defined in the compool.

RECRD

a. RECRD prints the CAT table.

b. No input parameters.

No output parameters.

c. No local variables.

d. No express variables

PCAT1  
OUT  
CATB  
CATC  
I1  
CATF  
J1

e. Procedures called

BTOD

f. Entered from

Main program

g. RECRD prints CATB, CATC, CATF, and I1. PCAT1, a Boolean variable, determines whether or not RECRD will be executed.

SCON

- a. On entry control, SCON either stores stati for status declared variables or searches for a given status in the STAT table VALU string reference scheme. Control variables are bookkept, discovered stati are converted to an STC integer in IDNT, and a found or not found Boolean indicator is set for immediate testing.

- b. Input parameters

INDEX = a STAT table index if a search is requested.

STORE = a Boolean indicator with the meanings:

true = 1 = store stati,

false = 0 = search for status in channel referenced by  
INDEX.

Output parameters

SCON = a Boolean set for immediate testing with the meanings:

1 = true = a status was found and converted,

0 = false = search in given channel was unsuccessful.

- c. No local variables

- d. Express variables

BRGTM	TIPE
IDNT	T2
NCHAR	T5
NSTAT	T12
NVAL	T13
NVALU	VALU
TEST	VPOS

- e. Procedures called

BTOD  
CAN  
ERROR  
GET2

- f. Entered from

IDEAL  
IMP



- g. If STORE = 1(true), store status; otherwise search.

Store function:

1. Set:

ERGIM(\$3\$) to NSTAT (next available channel in STAT table),

VPOS(\$NSTAT\$) to NVALU (next available byte in VALU)

NVAL(\$NSTAT\$) to the number of stati stored in VALU

NSTAT to NSTAT + 1

VALU to all stati after being stripped by the procedure CAN and the octal separator 77 being inserted following each status.

NVALU to the next available byte in VALU after the storage.

2. Use a call to the procedure GET2 to place the next status in IDNT until all stati are processed.
3. Use a call to the procedure CAN to check each constant in IDNT supplied by GET2, and process the constant.
4. Terminate when the part of speech in IDNT is not a status constant.

Search function:

1. IDNT contains the status of search.
2. INDEX contains the STAT table index referencing the stati to be searched.
3. VPOS(\$INDEX\$) contains the byte number of the first character of the first status in VALU.
4. NVAL(\$INDEX\$) contains the number of stati to search in VALU.
5. Count the first status of search as zero, the second as one, etc.
6. Analyze each character in VALU until a separator of octal 77 is reached.
7. Compare the status separated out with the status in IDNT. If not identical, bump the search counter, and if all stati in this channel are not compared, continue the search. If search is finished and unsuccessful, call procedure ERROR with input number 41. Set SCON to false and exit.

10 July 1962

107

TM-555/020/00

8. If search is successful, replace status in IDNT with an STC integer determined by search counter. Set SCON to true, NCHAR to the number of digits in the integer, T5 to NCHAR, and exit.

SRCH

- a. SRCH uses an access table called DISH with an index value supplied by the procedure HASH to determine the presence of a label, i.e., identifier, or constant defined in the dictionary, DICT. The label being searched is in IDNT. The results of the search are communicated with a Boolean indicator, and if found an index value to the defining DICT entry is supplied. House-keeping for reference and entry into IDNT is supplied. DISH items are maintained.

- b. No input parameters.

No output parameters.

- c. No local variables.

- d. Express variables

INUS	NOCN
DENT	NCHAR
SOCN	TEXT
NCHB	IDNS
FCHB	IDNT
FOND	T3
T4	NEXT

- e. No procedures called.

- f. Entered from

IDEAL  
POOL

- g. An entry index to table DISH is communicated from the procedure HASH by the express item T4. If the DISH channel so referenced is not occupied, then the label temporarily stored in IDNT is not defined by DICT. In this case, the Boolean FOND is set to false, i.e., zero, and the procedure is exited leaving the following settings:

T4 = DISH channel index (unchanged by SRCH),

T3 = the next available byte of IDNS from the simple item NEXT  
(unchanged by SRCH).

If the referenced DISH channel is in use, i.e., occupied, then two possibilities of a condition called "overflow" can exist. The index value supplied by HASH is equal to different labels "hashing" to the value, or

identical labels are defined by DICT more than once. Different domains of definition (e.g., the same label (identifier) is local to several procedures) or different scopes of definition (e.g., a statement label has the same name as a table) generate this latter type of overflow.

SRCH compares the label in IDNT against the label defined by DICT and stored in IDNS. The DISH item DENT provides an index to DICT, and the DICT items FCHB and NCHB provide a reference to IDNS. If the labels are not identical, the item SOCN of DISH is set to the next available overflow channel in the latter half of DISH. T<sub>4</sub> is then reset from SOCN and the procedure exits as described above. If the labels are identical, FOND is set to true, i.e., one, and if T<sub>3</sub> equals NEXT, T<sub>3</sub> is reset to FCHB from DICT. The procedure then exits as above.

If SRCH is entered with FOND equal to true, the procedure immediately resets for searching in the overflow portion of DISH prior to testing.

TSTIM

a. TSTIM reads the Hollerith input tape (HIN) containing the JOVIAL source program.

b. No input parameters.

No output parameters.

c. Local variables

AA

d. Express variables

CARD	HIN	TCD1
CATC	I1	TCD2
CATF	J1	TP
CDCL1	L1	TP1
DEBUG	LER	TSTR
DEFBP	QA	LBFN
EFIND	TCD	LBIN

e. Procedures called

ERROR

f. Entered from

Main program  
LOOK

- g. 1. If a defined identifier is being processed (DEFBP on), the procedure returns.
2. If an end of file has already been detected in the input (EFIND on), the current position in the card image is checked. If the entire card before the EOF has been processed, a TERM operator is placed in CAT, and error message is printed and processing is wound up in the normal manner. If the last card has not been entirely processed, return is made.
3. If the position in the card image (J1) is less than six bytes from the end of the card, another card must be read in.
4. If an EOF has previously been detected on the JOVIAL library tape (LBFN on), an error message is printed, and processing of the library is terminated, a TERM operator is placed in CAT and the program is completed.

10 July 1962

111  
(page 112 blank)

TM-555/020/00

5. If no EOP has been detected, the last six characters are moved from the last six CARD entries, to the first six, and the position in the card image (J1) is adjusted accordingly.
6. If the JOVIAL library tape is being read (LBIN on), a card is read from file LBR.
7. If not, a card is read from file HIN, and this card is written on the output listing file (DEBUG). The card is converted to STC, and the characters moved from their position in the input buffer (TSTR) to consecutive entries in item CARD. The procedure then returns.

10 July 1962

113

TM-555/020/00

E. ERROR MESSAGE DESCRIPTION

- 1 - START operator missing from JOVIAL source program.  
Detected by: main program, region JGOGIN
- 2 - Syntactical error.  
Detected by: main program, regions X2, X14, X3, X4, X5, X6
- \* 3 - Illegal character.  
Detected by: procedure LOOK
- \* 4 - Illegal COMM statement.  
Detected by: main program, region IO
- 5 - Procedure declaration found within procedure declaration.  
Detected by: main program, region PRIP
- 6 - Syntactical error.  
Detected by: main program, region IO
- 7 - Illegal \$\$.  
Detected by: procedure LOOK
- 8 - OPEN or SHUT not followed by INPUT or OUTPUT in statement.  
Detected by: main program, region IO
- \* 9 - Illegal DIRECT operator.  
Detected by: main program, region IO
- \*10 - Illegal ''.  
Detected by: main program, region PRIP
- 11 - DEFINE declaration not terminated by \$.  
Detected by: main program, region X3
- 12 - Binary point designator in A( ) (accumulator designator) not a constant (in ASSIGN statement).  
Detected by: main program, region PRAC
- 13 - Switch name missing in SWITCH declaration.  
Detected by: main program, region X26.
- 14 - "=" or "(" missing in SWITCH declaration.  
Detected by: main program, region X26
- 15 - Item name missing in item SWITCH declaration.  
Detected by: main program, region X26
- 16 - Procedure name missing in procedure declaration.  
Detected by: main program, region X27



- 17 - Close name missing in CLOSE declaration.  
Detected by: main program, region X28
- 18 - "\$" missing after close name in CLOSE declaration.  
Detected by: main program, region X28
- \*19 - No CAT entry available from IDEAL.  
Detected by: main program, region X3
- 20 - TERM operator missing or \$ after TERM operator missing.  
Detected by: procedure LOOK
- 21 - Comment terminated by \$.  
Detected by: procedure LOOK
- \*22 - Control transferred to zero point in a switch  
Detected by: procedure LOOK
- 23 - Uneven number of brackets encountered in current statement (includes  
"(", ")", "(\$", "\$)", "(/", "/)").
- 24 - Too many entries in the dictionary (DICT). Processing of the source  
program has been terminated.  
Detected by: main program, region X10
- 25 - Too many Overlay declarations (too many entries in table OVER).  
Processing of the source program has been terminated.  
Detected by: main program, region X10
- 26 - Too many characters of information in variable names, constants, etc.  
(variable IDNS contains too many characters). Processing of the  
sources program has been terminated.  
Detected by: main program, region X10
- 27 - Too many define declarations. Processing of the source program has  
been terminated.  
Detected by: main program, region X3
- 28 - Too many characters of information in define declarations (variable  
KD has overflowed) or too many characters in one definition and its  
associated nested defined labels. Processing of the source program  
has been terminated.  
Detected by: procedure LOOK
- 29 - HOLL or STC constant too large.  
Detected by: procedure LOOK
- 30 - Item, mode, or string declaration type code not recognized.  
Detected by: procedure IMP
- 31 - Item, mode, or string declaration number of bits or characters does  
not succeed type code for type A, D, H, or T.  
Detected by: procedure IMP

- 32 - Signed or unsigned indication in item, mode or string declaration is not recognized for type A or D.  
Detected by: procedure IMP
- 33 - Signed or unsigned indication in item, mode or string declaration is not recognized for type A or D.  
Detected by: procedure IMP
- 34 - Item declaration format is confused. Check the format preceding the "\$" character.  
Detected by: procedure IMP
- 35 - Simple item declaration does not have type code. Mode definition substituted.  
Detected by: procedure IDEAL
- 36 - (Not used)
- 37 - One of two constants used to specify range values in item declaration is missing.  
Detected by: procedure IMP
- 38 - Subscripted item declaration in table with specified packing has confused format preceding the "\$" character.  
Detected by: procedure IMP
- 39 - Simple item or mode declaration has data constant not consistent with variable type.  
Detected by: procedure IMP
- 40 - Status item, mode or string declaration has no stati.  
Detected by: procedure IMP
- 41 - Status constant assigned to variable was not found among variables' stati.  
Detected by: procedure SCON, IDEAL
- 42 - Referenced transfer point (statement, label, close, switch or compool address) is missing or undeclared procedure exists.  
Detected by: procedure IDEAL
- 43 - Subscript with non-subscripted variable or subscripted variable without subscript is referenced.  
Detected by: procedure IDEAL
- 44 - Overlay declaration has format in error or legal identifier not found in dictionary. Overlay accepted up to point of error.  
Detected by: procedure IDEAL

- 45 - File declaration has no stati.  
Detected by: procedure IDEAL
- 46 - File declaration has confused format.  
Detected by: procedure IDEAL
- 47 - Table declaration has confused format. Declaration passed over.  
Detected by: procedure IDEAL
- 48 - Like table not found for like table declaration.  
Detected by: procedure IDEAL
- 49 - Illegal subscripted variable identifier was generated as a result of  
a like table declaration. This variable omitted from table.  
Detected by: procedure IDEAL
- 50 - Table declaration confused between subscripted item declaration.  
Detected by: procedure IDEAL
- 51 - String or item declaration component of specified table declaration  
has confused format.  
Detected by: procedure IDEAL
- 52 - Constant type of pre-set table data is not consistent with other  
constants in same set, or not compatible with assigned variable.  
Detected by: procedure IDEAL
- 53 - Identifier does not succeed the formal word "ITEM" in simple item  
declaration.  
Detected by: procedure IDEAL
- 54 - Item declaration identifier already used for item, table, string or  
file in same area of scope.  
Detected by: procedure IDEAL
- 55 - Mode declaration not accepted, format illegal.  
Detected by: procedure IDEAL
- 56 - Array format error. Caution - array may have been abandoned.  
Detected by: procedure IDEAL
- 57 - Array dimensions are not pure unsigned integers.  
Detected by: procedure IDEAL
- 58 - Pre-set table data exceeds table capacity.  
Detected by: procedure IDEAL

- 59 - Constant has format error. Caution - reduced form may be junk.  
Detected by: procedure CAN
- 60 - Pre-set parameter constants assigned to an array with more than 3 dimensions is not currently implemented.  
Detected by: procedure IDEAL
- 61 - Pre-set parameter constants assigned to an array is excessive. The number of constants per row is greater than the number of columns.  
Detected by: procedure IDEAL
- 62 - Duplicate label - statement, switch or close within same area of scope.  
Detected by: procedure IDEAL
- 63 - Caution - mode definition used to define this non-subscripted variable.  
Detected by: procedure IDEAL
- 64 - Subscripted variable not previously declared - mode definition used to enable continued processing.  
Detected by: procedure IDEAL
- 65 - See 61. The number of constants per column is greater than the number of rows.  
Detected by: procedure IDEAL
- 66 - See 61. The number of constants per plane is greater than the number of planes.  
Detected by: procedure IDEAL
- 96 - More then 10 nested defined labels. (Label probably has been defined recursively.) Processing of the sources program has been terminated.  
Detected by: procedure LOOK
- 97 - Illegal single prime ('').  
Detected by: procedure LOOK
- 98 - Error in format of library file.  
Detected by: main program
- 99 - Trying to read past EOF on LIBR.  
Detected by: procedure TSTIM

P  
H  
A  
S  
E  
II

## DESCRIPTION OF PHASE II OF THE JOVIAL GENERATOR

## A. GENERAL DESCRIPTION

The second phase of the JOVIAL generator, hereafter referred to as Gen2, processes JOVIAL statements as they appear in their coded form as entries in the CAT table. It translates the statements into a machine-independent, parenthesis-free, prefixed operator notation called the intermediate language (IL). (For a detailed description of the intermediate language see FN-5066). The form of the intermediate language, the primary output of Gen2, is a table called the IL table, whose entries represent the original JOVIAL statements in an 'ordered' sequence of 'operator-operand-operand' functions (the ordering determined according to weights or priorities assigned the various JOVIAL operators and the nesting or parenthesizing of the statement).

All of the information required by the translators for the various types of JOVIAL statements can be supplied in entries of the IL table except for the FOR statement. Therefore, certain ancillary information associated with the FOR statements is supplied to the translators via the SRT table. An entry is made in the SRT table for each FOR statement supplying information including the range of the FOR statement in the IL table, the number of IL table entries required by its increment and test portions and the nesting level of the FOR statement. The SRT table, then, represents the second output of Gen2.

Two additional outputs supplied by the Gen2 include the total amount of temporary storage required by the program (N'TREG) and the number of generator-supplied labels added to the program (N'GLAB). (The generator supplies labels which are used as transfer points for relational statements, exit labels for procedures, labels for the top of FOR loops, etc.).

Gen2 processes first the main-part (MP) of a program and then its procedures. (This is the order in which the CAT table is arranged by Gen1). As the beginning of each procedure is encountered, it outputs the IL for the preceding procedure or MP. When the CAT table entries are exhausted or when the TERM bracketer is encountered, the last block of IL is output. At this time the SRT table is also output and formatted representations of the IL, SRT, and DRQ tables may be output (if requested). Gen2 has completed its function at this time.

The underlying flow of the processing carried on by Gen2 consists of the following:

- (1) Input - as CAT table entries are processed, new entries are input until an entire program has been processed.
- (2) Statement identification - the first entry of each statement is examined to determine the type of statement to be processed.
- (3) Statement processing - statements are processed according to their type (if an error is detected in a statement, the end of that statement is located so that the cycle always begins at the beginning of a new statement).
- (4) Delayed-processing - certain statements require two phases in their processing, one performed immediately, the second performed at a later time. The delayed-processing represents the second phase of processing. Therefore, following the processing of each statement it is necessary to perform any delayed-processing necessary.

Perhaps point (4) above should be discussed more fully. Certain statements require two phases in their processing. For example an IF statement. By definition, the statement succeeding an IF statement is to be executed only if the conditions specified in the IF statement are true. Therefore, it is necessary to label the end of the succeeding statement and to direct the flow of a program to this label, if the conditions specified in the IF statement are false. This label is called the 'false transfer point' label after the statement succeeding an IF statement represents the second phase, the delayed-processing, required by an IF statement. The order of operations becomes:

- (1) translate the IF statement into IL remembering its 'false transfer point';
- (2) translate the statement succeeding the IF statement into IL;
- (3) enter the 'false transfer point' label into the IL.

In addition to the IF statement, the following statement types require 2 phase processing: IFEITH, ORIF, FOR and procedure and close declarations. The ORIF statement requires for delayed-processing that the implied GOTO the end of the IFEITH statement and the false transfer point be entered into the IL. The IFEITH requires the same as the ORIF and also the final end label of the IFEITH to be entered. The FOR statement requires the bottom of a loop to be effected. The procedure and close declarations require that the end of the procedure or close be entered. To facilitate the delayed-processing, the DRQ table is used in a push-down manner to remember the type of delayed-processing required. Following the processing of each statement, the last entry in the DRQ table is

examined to see if it is time to do the delayed-processing required by a previous statement. The delayed-processing specified in a DRQ entry is performed after one simple or one compound statement has been processed since the time the entry was made (not including the statement which necessitated the DRQ entry).

Gen2 can be thought of as being composed of two main elements. Considering the elements in reverse order, the second element is comprised of the procedure ANCHR and its associated procedures. This element analyzes and converts into IL arithmetic and logical expressions. For a general description of the philosophy and operations of this procedure, see SP-127. The first element is comprised of the main program and its associated procedures. This is the control element of the program. It controls the input of CAT table entries; it recognizes the type of the statement to be processed and directs the processing thereof; and it recognizes when the processing is done and the program should exit. Many JOVIAL statements are not wholly arithmetic or logical in nature, for example, the FOR statement, and some are not arithmetic or logical at all, for example, the RETURN statement. The first element isolates these arithmetic and logical phrases from statements, directing ANCHR to process them, while processing the rest of these and all of the non-arithmetic and non-logical statements itself. The wholly arithmetic or logical statements are given over to ANCHR for processing.

The first element performs two additional functions. The first is concerned with the elimination of needless 'transfer' statements (GOTO, RETURN, TEST) and the optimizing of transfers implied by these statements via the alteration of the transfer points of IL GOTO and Relational operators.

An example of the elimination of needless transfer statements is as follows:

Consider: (1) the 2 statements

```
IF A LS B $  
GOTO LBL $
```

(2) the IL for the statement

```
A GQ B → FTP  
GOTO LBL  
Label FTP
```

Note - IL relationals imply a transfer if the conditions are true, while JOVIAL relationals imply a 'fall thru' to the next statement if the conditions are true.



Eliminate the transfer statement by altering the IL.

A LS B → LBL

An example of the optimizing of transfers is as follows:

Consider: (1) the statements

IF A LS B \$  
C = D \$  
GOTO LBL \$

(2) the IL for the statements

A GQ B → FTP  
C = D  
Label FTP  
GOTO LBL

Optimize the transfer by altering the IL:

A GQ B → LBL  
C = D  
GOTO LBL

An example of elimination and optimizing is:

Consider: (1) the statements

IF A LS B or  
C EQ D \$  
GOTO LBL \$

(2) the IL for the statements

A LS B → TTP  
C NQ D → FTP  
Label TTP  
GOTO LBL  
Label FTP

Optimize and elimination by:

A LS B → LBL  
C EQ D → LBL

The second additional function performed by the first element of Gen2 is concerned with the determination of the minimum amount of temporary storage necessary for each procedure. This determination involves an analysis of the temporary storage required by the procedures which call the given procedure. The minimum amount represents the starting level at which the temporary storage must commence for a given procedure. The actual analysis is performed

10 July 1962

123  
(page 124 blank)

TM-555/020/00

by procedure WORST and its results are recorded in the DICT entry associated with each procedure. It also determines the total amount of temporary storage required by the entire program during the analysis.

10 July 1962

125

TM-555/020/00

## B. GLOSSARY

Item, Table, Procedure, Parameter, Switch, and File Descriptions



10 July 1962

127

TM-555/020

ASNTP	Item	Local	Used by: ANCHR
	Indicates when a nested bead expression has just been assigned to temporary storage of result (the first or left operand of the IL assignment specifies temp 0 (ILD = 3, ILF = 0)).		
	1 = nested bead expression just assigned to result 0 = other than nested bead just processed		
ASPAC	Item	Express	Used by: ANCHR, SS
	Specifies the number of entries +1 in the CAT table between the ABS modifier and its right parenthesis.		
ATTP	Item	Express	Used by: ANCHR, XF
	Contains the generated label number of the 'true transfer point' for a Boolean assignment statement.		
AUI	Item	Local	Used by: ANCHR
	Indicates whether a unary minus or an ABS modifier is being processed		
	1 = ABS 0 = unary minus		
AVPOS	Subscripted item from table SS	Express	Used by: ANCHR, SS
	Contains the CAT table entry number describing the ABS modifier the subscript expression.		

B1	Item	Express	Used by: MP
	Indicates whether a directed TEST statement is being processed (TEST Q \$).		
	1 = directed TEST		
	0 = non-directed TEST		
B2	Item	Express	Used by: MP
	Indicates if a TEST statement was used in the range of some FOR statement.		
	1 = used in the range of some FOR		
	0 = used outside the range of a FOR		
BASE	Item	Local	Used by: ANCHR
	A 'push-down' type item used to count brackets (incremented by 20 for each left bracket, decremented by 20 for each right bracket).		
BCAT	File	Express	Used by: MP
	Contains the binary CAT table as output by Genl representing the main body of a program.		
BCATP	File	Express	Used by: MP
	Contains the binary CAT table as output by Genl representing the procedures of a program.		
BD	Item	Local	Used by: BDSV
	Indicates whether a bead or subscript expression is being processed by BDSV.		
	0 = subscript expression		
	1 = BYTE expression		
	2 = BIT expression		
BDBSE	Subscripted item from table BEADS	Express	Used by: ANCHR
	Contains the value of the base (BASE) at the beginning of each beaded expression.		
BDCNT	Item	Local	Used by: ANCHR
	Specifies the number of entries in table BEADS.		

**TM-555/020/00**

0 = subscripted expression  
1 = BYTE expression  
2 = BIT expression

1 = BYTE  
2 = BIT

BEADS	Table	Express	Used by:	ANCHR
	This table is used in a push-down manner and contains information used in the recognition of the parenthesis required in a beaded expression.			

Example: `BYTE [ A,B ] {VAR}`

**Used to recognize parenthesis 1 & 2**

EKI	Item	Local	Used by: SRCHR
	Push-down type item used to count bracketers (incremented by left bracketer, decremented by right bracketers).		

BNWS	Item	Express	Used by:
			ANCHR, PUTIN BDSV

Indicates whether a temporary storage assignment should be changed back to specify 'result' (Temp 0)

```
1 = change back to result
0 = do not change
```

<b>BOOL</b>	Item	Express	Used by: MP
	Indicates when a FOR ALL type statement is being processed (FOR I = ALL (TABLE'NAME) \$		
	1 = FOR ALL statement 0 = non-FOR ALL statement		
<b>BOOL1</b>	Item	Express	Used by: MP
	Indicates whether a multiple factor FOR statement is being processed.		
	1 = multiple factor FOR statement 0 = 1-factor FOR statement		
<b>BOOL3</b>	Item	Express	Used by: MP
	Meaning of item depends on use.		
	1st use: Indicates whether a 2- or 3-factor FOR statement is being processed		
	1 = 2-factor FOR 0 = 3-factor FOR		
	2nd use: Indicates whether the 3rd factor was terminated by a \$		
	1 = 3rd factor terminated by a \$ 0 = 3rd factor not terminated by a \$		
<b>EPDICT</b>	Subscripted item from table COMMEN	Express	Used by: -
	Indicates whether a formatted listing of the IL table entries has been requested of the translator.		
<b>EPILT</b>	Subscripted item from table COMMEN	Express	Used by: -
	Indicates whether a formatted listing of the IL table entries has been requested of the translator.		
<b>BPTDT</b>	Subscripted item from table COMMEN	Express	Used by: -
	Indicates whether a formatted listing of the TDICT table entries has been requested of the translator.		
<b>BRAKS</b>	Switch		Activated in: MP
	Sensitive to the form (CATF) of an entry with a class of bracketter (CATC EQ 10).		





CALL	Item	Express Used by:
		MP, WORST, ANCHR
	Contains the DICT entry number of the last procedure call processed.	
CAT	Table	Contains items: CATA, CATB, CATC, CATE, CATF Express Used by: Entire program.
	Contains the object program in coded notation.	
CATA	Subscripted item from table CAT	Express Used by: Entire program
	Specifies the sign (if relevent) associated with the entry.	
	1 = Minus 0 = Plus	
CATB	Subscripted item from table CAT	Express Used by: Entire program
	Specifies the modifier (if relevent) associated with the entry.	
	0 = Null            4 = CHAR            8 = POS 1 = NENT           5 = MANT           9 = - 2 = ENT            6 = ABS            10 = ODD 3 = NWDSEN        7 = ALL	
CATC	Subscripted item from table CAT	Express Used by: Entire program
	Specifies the 'class' of the information described in the entry (see the description of CATF for the names of the various classes).	
CATE	Subscripted item from table CAT	Express Used by: Entire program
	Contains the level (see description of procedure ANCHR) associated with entries describing operators; Specifies the number of cards associated with an entry of class 'Direct code' (CATC EQ 17).	

CATF      Subscripted item from table CAT      Express      Used by: Entire program

Specifies the 'form' or sub-class of the information described in the entry; the actual meaning is dependent on the 'class' (CATC) of the entry.

Class (CATC)	Form (CATF)
0 = Null	Null
1 = Statement label	Dictionary entry number
2 = File label	Dictionary entry number
3 = Constant	Dictionary entry number
4 = Simple item	Dictionary entry number
5 = Subscripted item	Dictionary entry number
6 = Procedure label	Dictionary entry number
7 = Subscript	Subscript number ( $1 \leq n \leq 26$ )
8 = Sequential operator	0 = IF      4 = FOR
	1 = IFEITH      5 = RETURN
	2 = ORIF      6 = STOP
	3 = GOTO      7 = TEST
9 = Separator	0 = .      3 = ==
	1 = ,      4 = \$
	2 = =      5 = ...
10 = Bracketer	0 = (      6 = START
	1 = )      7 = TERM
	2 = [      8 = [ (associated
	3 = ]      with a nested Bead)
	4 = BEGIN      9 = Double prime
	5 = END
	10 = ( (associated with a Bead)
	11 = ) (associated with a Bead)
	12 = [ (associated with a nested
	subscript expression of
	non-standard form)
11 = Logical operator	0 = AND      1 = OR      2 = NOT
12 = Relational operator	0 = LS      2 = LQ      4 = NQ
	1 = EQ      3 = GR      5 = GQ
13 = Arithmetic operator	0 = LS      3 = /      6 = Unary-
	1 = -      4 = **      7 = ABS
	2 = *      5 = 'OF' (always
	included following an entry
	describing a procedure call)

14 = Declaration

0 = Switch	5 = Array
1 = Procedure	6 = Overlay
2 = Close	7 = File
3 = Item	8 = Mode
4 = Table	9 = Define
	10 = String

15 = Input-Output

0 = Input	3 = Open Output
1 = Output	4 = Shut Input
2 = Open Input	5 = Shut Output

16 = Bead

0 = BYTE	2 = Bead
1 = BIT	(String)

17 = Direct code

First card number (The number of cards is specified in CATE).

18 = Temporary storage

Specifies the specific temporary storage 'register' (0 = result).

19 = Generated label

Specifies the generated label number.

20 = Direct code assignment operand  
(A( ) )

Specifies the bits to the right of the binary point (Unsigned); all ones for floating point.

CATPS

Input Parameter

Used by: PUTIN

Specifies the starting CAT table entry of the expression to be processed.

CAWL

Subscripted item from table LWC

Local

Used by: WORST

Contains the DICT entry number of the procedure called.

CCNT

Item

Local

Used by: ANCHR

Specifies the number of entries in table COMMA.

- CHL1**      Subscripted item from table DICT    Express    Used by: -
- If CLAS EQ constant and SIND EQ 1 and the nearest preceeding non-constant entry is a subscripted item or array:  
        Specifies the number of constants (in this row).
- If CLAS EQ item, subscripted item or array and RNGI EQ 1:  
        Specifies the first character byte in item IDNS of the specified range values.
- If CLAS EQ string:  
        Specifies the index increment to the next word of this entry containing beads.
- If CLAS EQ table:  
        Specifies the number of entries in this table.
- If CLAS EQ procedure:  
        Specifies the DICT entry number of the first variable associated with the procedure (0 = undeclared procedure).
- If CLAS EQ file:  
        Specifies the maximum number of bits/bytes.
- If CLAS EQ item switch:  
        Specifies the DICT entry number of the associated item.
- CHL2**      Subscripted item from table DICT    Express    Used by: -
- If CLAS EQ procedure:  
        Specifies the number of DICT entries associated with the procedure.
- If CLAS EQ file:  
        Specifies the first byte in item IDNS of the hardware label.
- If CLAS EQ item/numeric switch:  
        Specifies the number of switch points.
- If CLAS EQ array:  
        Specifies the DIMZ entry number containing the first dimension of the array (successive dimensions are in successive entries).
- If CLAS EQ table and DEFN NQ 2:  
        Specifies the number of associated DICT entries.
- If CLAS EQ string:  
        Specifies the number of beads per word.
- If CLAS EQ constant and SIND EQ 1 and the nearest preceeding non-constant entry is array:  
        Specifies the DICT entry of the associated array.

**CIND** Subscripted item from table COMMA Express Used by: ANCHR  
 Indicates if the last bracketer encountered was a left parenthesis or left bracket.  
 1 = left bracket (therefore when a comma is encountered, it must terminate a bead or subscript expression).  
 0 = left parenthesis (therefore when a comma is encountered, it must be internal to a procedure call).

**CLAS** Subscripted item from table DICT Express Used by: MP, DESIG, ANCHR  
 Specifies the class of the entry.  
 0 = Null  
 1 = Statement label  
 2 = File  
 3 = Constant  
 4 = Simple item  
 5 = Subscripted item  
 6 = Procedure function  
 7 = Table  
 8 = Array  
 9 = Item switch  
 10 = Numeric switch  
 11 = Close  
 12 = Compool  
 13 = String variable

**CLASS** Switch Activated in: MP  
 Sensitive to the class (CATC) of an entry.

**CLASS** Switch Activated in: PUTIN  
 Sensitive to the class (CATC) of an entry.

**CLSS** Switch Activated in: ANCHR  
 Sensitive to the class (CATC) of an entry.

**CLSW** Switch Activated in: SSFRO  
 Sensitive to the class (CATC) of an entry.

**COLSDATA** Subscripted item from table COMMTN Express Used by: -  
 Specifies the number of columns on the input cards which are considered to contain data of the object program.

**COLSIN** Subscripted item from table COMMTN Express Used by: -  
 Specifies the number of columns on the input cards which are read.

**COMAS** Item Local Used by: BDSV  
 Contains a count of the number of commas encountered by BDSV.

COMI	Item	Local	Used by: ANCHR
------	------	-------	----------------

Indicates whether a comma terminates a subscript expression.

1 = comma terminated the expression  
0 = right bracket terminated the expression

COMI	Output Parameter		Used by: SSPRO
------	------------------	--	----------------

Indicates whether the expression terminated with a ','.

1 = terminated with a comma  
0 = terminated with a non-comma

COMMA	Table	Express	Used by: ANCHR
-------	-------	---------	----------------

This table is used in a push-down manner in the analysis of commas to determine when commas terminate subscript or head expressions and when they terminate parameters in procedure calls.

COMMTN	Table	Contains items:	PCATI, PSRT, PDRQ, PIL, PDCTY, PCAT2, BPDICT, BPILT, BPTDT, IDENT, COMPOL, ORIGIN, STOP'IFG, STOP'IFT, NCIDNS, N'TREG, N'GLAB, N'CONX, GEN'ERR, TRAN'ERR, COLSIN, COLSDATA
--------	-------	-----------------	--

	Express	Used by: MP, ANCHR, WORST, ERROR
--	---------	----------------------------------

Communication table linking the central program or system, the generator and the translator.

COMPOL	Subscripted item from table COMMTN	Express	Used by: -
--------	------------------------------------	---------	------------

Specifies the ident of the compool to be used in the compilation.

CONTRL	Table	Contains item: Control	Express	Used by: (Dummy table)
--------	-------	------------------------	---------	------------------------

A dummy table to allocate space for the system control program.

CONTROL	Subscripted item from table CONTRL	Express	Used by: (Dummy table)
---------	------------------------------------	---------	------------------------

Dummy item.

CTPOS	Item	Local	Used by: PUTIN
-------	------	-------	----------------

Specifies the CAT table entry currently being processed by PUTIN

**DEBUG**      **File**      **Express**      **Used by: PRMT**  
 Contains the formatted error messages and table listings.

**DEFN**      **Subscripted item from table DICT**      **Express**      **Used by: MP, ANCHR**  
 Specifies the definition of the entry.

0 = Null      3 = Local  
 1 = Express      4 = Formal procedure input parameter  
 2 = Compool      5 = Formal procedure output parameter

**DELE**      **Input Parameter**      **Used by: BDSV**  
 Indicates whether the CAT table entries examined are to be cleared.

1 = do not clear  
 0 = clear

**DELET**      **Input Parameter**      **Used by: PUTIN**  
 Indicates whether the entries examined are to be cleared.

1 = do not clear  
 0 = clear

**DESIG**      **Procedure**      **Used by: MP**  
 Analyzes and converts into IL switch point expressions or GOTO expressions.

**DICT**      **Table**      **Contains items: PDAT, FCHB, LCPL, TFRM, RNGI, SIZE, CHL2, RORT, FPNM, DEFN, TYPV, PACK, TREGN, BRGT, SIND, FBIT, CHL1, CLAS, NCHB, DIMN**  
**Express**      **Used by: Entire program**  
 Contains complete description of the constants, tables, items etc. of the source program as derived from their declarations.

**DIMN**      **Subscripted item from table DICT**      **Express**      **Used by: -**  
 If CLAS EQ file:  
     Specifies the number of characters in item IDNS of the hardware label  
 If CLAS EQ constant and SIND EQ 1 and the nearest non-constant preceding entry is array:  
     Specifies the index (relative to 0) of the plane (3rd dimension) of the set of data  
 If CLAS EQ array:  
     Specifies the number of array dimensions



If CLAS EQ subscripted item or string:

Specifies the DICT entry number of the associated label.

**DNDRQ**      Item                              Express      Used by: MP, INSQ  
Specifies the number of entries in the temporary push-down queue (entries 1 and 2 of table DRQ).

**DONE**      Item                              Express      Used by: MP  
Indicated when an end-of-file has been encountered reading file BCATP or when a CAT entry describing a TERM has been encountered.

1 = encountered EOF or a TERM  
0 = did not encounter an EOF or a TERM

**DRQ**      Table      Contains items: DRQA, DRQB, DRQC, DRQE, DRQF, DRQI, DRQJ  
Express      Used by: MP, INSQ, IFO  
This table is used in a push-down manner and contains information related to the delayed processing required by FOR, IF, IFEITH, ORIF, PROC, and CLOSE statements. Entries 1 and 2 of this table are reserved as a temporary push-down queue to facilitate the processing of (1) successive statements requiring delayed-processing and (2) statements requiring delayed-processing located just prior to the END terminating a compound statement. Entries are first made in the temporary queue and then moved into the normal DRQ at the appropriate time.

**DRQA**      Subscripted item from table DRQ      Express      Used by: MP, INSQ  
Specifies the type of statement this entry refers to.  
0 = FOR statement                              3 = ORIF statement  
1 = PROC/CLOSE statement                      4 = IFEITH statement  
2 = IF statement

**DRQB**      Subscripted item from table DRQ      Express      Used by: MP, INSQ  
Contains the value of the statement parenthesis counter (STPC) when the entry was made.

**DRQC**      Subscripted item from table DRQ      Express      Used by: MP, INSQ  
Contains the value of the statement counter (STMC) when the entry was made.

**DRQD**      Subscribed item from table DRQ      Express    Used by: MP, INSQ  
Contents varies according to the type of entry (DRQA).

DRQA EQ 0 (FOR): Contains the SRT entry number associated with the FOR.

DRQA EQ 1 (PROC/CLOSE): Contains the IL entry number of the procedure entrance in IL.

DRQA EQ 2,3,4 (IF, IFEITH, ORIF): Contains the false transfer point generated label number.

**DRQE**      Subscribed item from table DRQ      Express    Used by: MP, INSQ  
Contains a count of the number of FOR statements this entry refers to (important only for parallel FOR's).

**DRQF**      Subscribed item from table DRQ      Express    Used by: MP, INSQ  
Indicates when this entry references a 2 or 3 factor FOR or when a 2 or 3 factor FOR statement has been encountered in a parallel group.

**DRQI**      Subscribed item from table DRQ      Express    Used by: MP, INSQ, IFO  
Contents varies depending on type of entry (DRQA).

DRQA EQ 0 (FOR): Specifies the generated label number for the top of the loop.

DRQA EQ 2,3,4, (IF, IFEITH, ORIF): Specifies the first IL entry number required for the statement.

**DRQJ**      Subscribed item from table DRQ      Express    Used by: MP, INSQ, IFO  
Specifies the last IL entry number required for an IF, IFEITH or ORIF statement.

**DRQD** Subscripted item from table DRQ Express Used by: MP, INSQ  
Contents varies according to the type of entry (DRQA).  
DRQA EQ 0 (FOR): Contains the SRT entry number associated with the FOR.  
DRQA EQ 1 (PROC/CLOSE): Contains the IL entry number of the procedure entrance in IL.  
DRQA EQ 2,3,4 (IF, IFEITH, ORIF): Contains the false transfer point generated label number.

**DRQE** Subscripted item from table DRQ Express Used by: MP, INSQ  
Contains a count of the number of FOR statements this entry refers to (important only for parallel FOR's).

**DRQF** Subscripted item from table DRQ Express Used by: MP, INSQ  
Indicates when this entry references a 2 or 3 factor FOR or when a 2 or 3 factor FOR statement has been encountered in a parallel group.

**DRQI** Subscripted item from table DRQ Express Used by: MP, INSQ, IFO  
Contents varies depending on type of entry (DRQA).  
DRQA EQ 0 (FOR): Specifies the generated label number for the top of the loop.  
DRQA EQ 2,3,4, (IF, IFEITH, ORIF): Specifies the first IL entry number required for the statement.

**DRQJ** Subscripted item from table DRQ Express Used by: MP, INSQ, IFO  
Specifies the last IL entry number required for an IF, IFEITH or ORIF statement.

**EEEE**      Output Parameter      Used by: **BDSV**  
Specifies the error exit for the procedure.

**ENDS**      Switch      Activated in: **MP**  
Sensitive to the type of delayed-processing (**DRQA**) required.

**ERAS**      Item      Local      Used by: **ANCHR**  
Indicates whether entries are to be cleared in the CAT table as they are translated into IL entries.  
1 = do not erase entry  
0 = erase entry

**ERCNT**      Item      Express      Used by: **ERROR**  
Set to zero initially. Count of entries to **ERROR** procedure.

**ERROR**      Procedure      Used by: **MB, FACT, DESIGN, LENGTH, WORST, ANCHR, SSFRO, POSOD, PUTIN, BDSV, XFRPT**  
Composes an error message into item **IMAGE**.

**EXCP**      Item      Local      Used by: **WORST**  
Indicates when one of **WORST**'s local tables has been exceeded.  
1 = local table exceeded  
0 = local table not exceeded



GEN'ERR Subscribed item from table COMMTN Express Used by: ERROR

Indicates whether errors were discovered in the object program by the generator.

GLAB	Item	Express	Used by: MP, ANCHR, DESIG
------	------	---------	---------------------------

**Specifies the number of generated labels currently assigned (synonymous with LBGR).**

GO	Item	Local	Used by: OPTM
----	------	-------	---------------

Indicates whether an IL GTO (ILO EQ 15) entry was encountered while trying to optimize.

1 = GTO entry encountered

0 = non-GTO entry encountered

GOI	Item	Local	Used by: OPTM
-----	------	-------	---------------

Indicates whether an IL GTO (ILO EQ 15) or dummy-TST (ILO EQ 25, ILE EQ 0) entry was encountered while trying to optimize.

1 = GTO or Dummy-TST encountered

0 = non GTO or non-Dummy-TST encountered

10 July 1962

144

TM-555/020/00

HEDSW	Switch	Activated in: PRTHD
	Sensitive to the number of the heading to be printed (PII).	
HL	Output Parameter	Used by: HOLLS
	Contains those characters of TC which were converted.	
HOLLS	Procedure	Used by: PRMT
	Converts standard transmission characters into Hollerith code.	

**IDENT** Subscripted item from table **COMMTN** Express Used by: -  
Specifies the ident of the object program.

**IDNS** Item Express Used by: Entire Program  
Contains all identifiers, literal constants, preset data constants, range constants, define labels and hardware labels in **STC** code.

**IFEIX** Item Express Used by: MP, **INSQ**  
Contains the generated label number of the bottom of the **IFEITH** statement currently being processed.

**IFO** Procedure Used by: MP  
Generates **IL** 'information' type entries which supply the boundary **IL** entry numbers of a relational statement referring to the label preceding the 'information' entries.

**IFOI** Item Local Used by: **OPTM**  
Indicates when an **IFO** **IL** entry (**ILO** EQ 34) and its associated **IL** operand entry must not be cleared.  
1 = do not clear  
0 = clear

**IL** Table Contains items: **ILC**, **ILA**, **ILH**, **ILB**, **ILO**, **ILD**, **ILF**, **ILE**, **ILR**, **ILI**, **ILY**, **ILZ**  
Express Used by: MP, **FACT**, **DESIG**, **OPND**, **OPTM**, **STLE**, **IFO**, **PRTIL**, **ILDF**, **ANCHR**, **XFRPT**, **PUTIN**, **BDSV**  
Contains the intermediate language representation of a program.

**ILA** Subscripted item from table **IL** Express Used by: (see table description)  
Meaningful only if **ILC** EQ 1, 2, 3 (operand, subscript or bead forms).  
Indicates the sign of the form.  
1 = Minus  
0 = Plus



**ILB**      Subscripted item from table IL      Express      Used by: (see table description)

Meaningful only in operand, subscript or bead form entries.

(ILC EQ 1,2,3)

Specifies the modifier of the entry.

0 = Null	4 = CHAR	8 = POS
1 = NEWT	5 = MANT	9 = Null
2 = ENT	6 = ABS	10 = ODD
3 = NWDSEN	7 = ALL	

**ILC**      Subscripted item from table IL      Express      Used by: (see table description)

Specifies the form or type of entry.

0 = operator form	2 = subscript form
1 = operand form	3 = bead form

**ILD**      Subscripted item from table IL      Express      Used by: (see table description)

Specifies the code or type of information in item ILF.

0 = ILF contains a DICT entry number
1 = ILF contains a generated label number
2 = ILF contains a subscript number
3 = ILF contains a temporary storage number

**ILDF**      Procedure      Used by: PRFIL

Converts the code (ILD) and channel (ILF) of an IL entry into output format.

**ILE**      Subscripted item from table IL      Express      Used by: (see table description)

Value depends on the class of the entry (ILC)

ILC EQ 0 (operator):

- (1) if ILO EQ 26 (TST): contains the generated label number as a transfer point to the top of a FOR loop.
- (2) for all other values of ILO: contains the statement number relative to the last label.

ILC EQ 1 (operand): If the operand is under a SWN operator entry: contains the DICT entry number of the constant associated with a switch point.

If this is the first operand entry following a PCL entry: contains a count of the number of parameters in the procedure call.

ILC EQ 2, 3 (subscript or bead): contains the DICT entry number of the constant increment of the expression.

ILF        Subscripted item from table IL        Express        Used by: (see table description)

Value depends on the code of the entry as specified by ILD.

ILFF       Input Parameter        Used by: ILDF

Specifies the IL entry number of the entry to be processed.

ILH        Subscripted item from table IL        Express        Used by: (see table description)

Indicates the type of bead.

1 = BIT  
0 = BYTE

ILI        Subscripted item from table IL        Express        Used by: (see table description)

Meaningful only in an information operator entry or the operand entry following an information operator entry.

Specifies an IL entry number. If in an operator entry - contains the entry number of the 1st entry of the relational statement with which the label entry preceding the information entry is associated. If in an operand - contains the last entry number for the relational statement.

ILLOC       Item        Local        Used by: ANCHR

Contains the IL table entry of the first (or left) operand of a nested bead assignment statement.

**ILO**      Subscripted item from table IL      Express      Used by: (see table description)

Meaningful only in operator entries (ILC EQ 0)

Specifies the operator in the entry

0 = Null	18 = Start
1 = LS	19 = Term
2 = EQ	20 = Stop
3 = LQ	21 = Assign (=)
4 = GR	22 = Exchange (==)
5 = NQ	23 = Load (1st FOR factor)
6 = GQ	24 = Increment (2nd FOR factor)
7 = +	25 = Test (3rd FOR factor)
8 = -	26 = Nester
9 = *	27 = Direct code
10 = /	28 = Input
11 = **	29 = Output
12 = Procedure call	30 = Open input
13 = Procedure entrance	31 = Open output
14 = Procedure exit	32 = Shut input
15 = GOTO	33 = Shut output
16 = Label	34 = Information
17 = Switch name	

**ILPOS**      Input Parameter      Used by: XFRPT

Specifies the IL table entry of the last relational operator entered into the IL.

**ILR**      Subscripted item from table IL      Express      Used by: (see table description)

Meaning depends on class of entry (ILC)

ILC EQ 0 and ILO EQ LBL: 1 - this label is at the top of a FOR loop  
0 - normal label

ILC EQ 1 and if this is the first operand following a relational operator (1 ILO 6): 1 - sense of operator was reversed  
0 - sense of operator not reversed

If ILC EQ 1 and the operands represent the parameters from a procedure call: 1 - output parameter  
0 - input parameter

ILY      Subscripted item from table IL      Express    Used by: (see table description)

Meaningful only if ILC EQ 0 and ILO EQ <sup>25</sup>26.  
Indicates if this is a dummy TST entry.

1 = dummy TST  
0 = non-dummy TST

ILZ      Subscripted item from table IL      Express    Used by: (see table description).

An abbreviated form of ILF (has fewer bits).

IMAGE    Item      Express    Used by: MP, ERROR, PRNT, PRTIL, PPTCT, ILDF, PRTHD, PRTSR, PRTQ, NTOI

Item output by PRNT which contains formatted expressions to be printed.

INSQ    Procedure      Used by: MP

Makes entries into the DRQ table depending on the type of delayed-processing required by certain statements.

IOFI    Item      Local      Used by: ANCHR

Indicates whether an input or an output parameter of a procedure call is being processed.

1 = output parameter  
0 = input parameter

IOS    Item      Express    Used by: ANCHR

Indicates that input/output statement is being processed.

10 July 1962

150

TM-555/020/00

JOB Switch

Activated in: WORST

Sensitive to the <sup>Type</sup>~~type~~ (AA) of processing to be performed by the procedure.

LAST	Item	Express	Used by: MP, ANCHR, LLOC, LVLCP
	Procedure ANCHR terminates its analysis of CAT table entries at the entry specified in this item.		
LBGR	Item	Express	Used by: MP, ANCHR
	Specifies the number of generated labels currently assigned (synonomous with GLAB).		
LBLOC	Item	Local	Used by: ANCHR
	Contains the CAT table entry describing the left bracket of a nested subscript expression of standard form.		
LBPOS	Subscripted item from table SS	Express	Used by: ANCHR, SSFRO
	Contains the CAT table entry number describing the left bracket of this subscript expression.		
LCAT	Item	Local	Used by: BDSV
	Specifies the CAT table entry currently being processed by BDSV.		
LEFT	Item	Express	Used by: ANCHR, LVLCP
	Specifies the CAT table entry number of the operator immediately to the left of the 'anchor point'.		
LEV	Output Parameter		Used by: LVLCP
	Indicates the outcome in the attempt to locate an 'anchor point'.		
	0 = no 'anchor point' determined		
	1 = 'anchor point' determined		
	2 = end-of-statement encountered		
	3 = no level encountered in the statement		
LEVEL	Item	Local	Used by: ANCHR
	Indicates whether an 'anchor point' has been determined.		
	0 = no 'anchor point'; assign temporary storage if required		
	1 = 'anchor point' determined, process operator		
	2 = an end-of-statement 'anchor point' was determined		
	3 = no 'anchor point', unable to locate an operator left of the current 'anchor point' within the expression.		

LIL	Item	Local	Used by: BDSV
	Specifies the IL entry at which the subscript or bead expression is to be located.		
LLCAT	Input Parameter		Used by: BDSV
	Specifies the starting CAT table entry number of the expression to be processed.		
LLO	Output Parameter		Used by: LLOC
	Specifies the entry number in the CAT table of the entry at which the scan was terminated.		
LLOC	Input Parameter		Used by: ANCHR, LVLCP
	Performs a backward scan thru the CAT table searching for an entry having a level (CATE EQ 0).		
LLOC	Input Parameter		Used by: XFRPT
	Specifies the CAT table entry last located by procedure LLOC.		
LN	Item	Local	Used by: ANCHR
	Specifies the number of entries in table LOGIC.		
LNAME	Item	Express	Used by: MP, ERROR
	Contains the DICT entry number of the last statement label encountered.		
LNQH	Output Parameter		Used by: LNQH
	Indicates the number of entries scanned by the procedure.		
LNQH	Procedure		Used by: MP, DESIG
	Performs a scan of the CAT table looking for a comma or a \$.		
LOC	Item	Express	Used by: MP
	Contains the IL entry number of the first entry used by the statement currently being processed.		
LOC	Item	Local	Used by: SRCHL
	Specifies the CAT table entry currently being examined by procedure SRCHL.		

LOC	Item	Local	Used by: SRCHR
	Specifies the CAT table entry currently being examined by procedure SRCHR.		
LOCA	Subscripted item from table LWC	Local	Used by: WORST
	Contains the LWC entry number describing the first procedure call made by the procedure specified in the term CAWL of this entry.		
LOCC	Input Parameter		Used by: SRCHR
	Specifies the starting CAT table entry number for the scan.		
LOCC	Input Parameter		Use by: SRCHR
	Specifies the starting CAT table number for the scan.		
LOG	Table	Contains item: LOGIC	Express Used by: ANCHR
	This table is used in a push-down manner and contains information used in the analysis of the NOT relational operator (indicates when relational operators are under the influence of a NOT).		
	Stepped by: each NOT		
	Reduced by: if the level in an entry is less than the current base on relational operators, Boolean items or right parentheses.		
LOGIC	Subscripted item from table LOG	Express	Used by: ANCHR
	Contains the base + 20 at the time a NOT is encountered. Is used to determine when relational and Boolean expressions are under the influence of a NOT.		
LOP	Switch		Activated in: ANCHR
	Sensitive to the form (CATF) of an entry with a class of logical operator (CATC EQ 11).		
LOPR	Subscripted item from table LWC	Local	Used by: WORST
	Contains the PWS entry number of the PWS entry describing the procedure specified in the term CAWL of this entry.		
LOWS	Subscripted item from table LWC	Local	Used by: WORST
	Specifies the amount of temporary storage required prior to making the procedure call.		



10 July 1962

154

TM-555/020/00

LPLUS	Item	Express	Used by: MP, ERROR
	Contains a count of the number -1 of statements processed since the last statement label was encountered.		
LSRCH	Output Parameter		Used by: SRCHL
	Indicates whether the scan was successful.		
	1 = successful		
	0 = unsuccessful		
LVLCP	Procedure		Used by: ANCHR
	Attempts to locate an anchor-point in the CAT table.		
LWC	Table	Contains items: LOCA, LOPR, CAWL, PROW, LOWS	
		Local	Used by: WORST
	Contains information about the amount of temporary storage required at procedure calls.		

10 July 1962

155

TM-555/020/00

MAXWS	Item	Express	Used by: MP, ANCHR, WORST
	Specifies the maximum amount of temporary storage required by a program.		
MID	Item	Express	Used by: ANCHR
	Specifies the CAT table entry of the last operator entered into the IL.		
MYNUS	Item	Express	Used by: MP, DESIG
	Indicate whether an item switch point has a negative constant associated with it.		
	1 = negative constant		
	0 = positive constant		

N'CONX	Subscripted item from table COMMTN Express	Used by: -
	Specifies the number of constants in the object program.	
N'GLAB	Subscripted item from table COMMTN Express	Used by: MP
	Specifies the number of labels supplied by the generator.	
N'TREG	Subscripted item from table COMMTN Express	Used by: WORST
	Specifies the amount of temporary storage required by the object program.	
NCHB	Subscripted item from table DICT Express	Used by: WTOI
	If CLAS EQ constant and SIND EQ 1 and the nearest preceding non-constant entry is a subscripted item or array: Specifies the number of bytes in the first constant set (excluding the separator character).	
	For all other classes: Specifies the number of character bytes in item IDNS of the identifier associated with the entry.	
NCIDNS	Subscripted item from table COMMTN Express	Used by: -
	Specifies the number of characters of information contained in item IDNS.	
NDRQ	Item	Express Used by: MP, INSQ
	Specifies the number of entries -1 in the DRQ table.	
NEST	Item	Express Used by: MP, INSQ
	Contains a count of the number of subscripts currently active.	
NIL	Item	Express Used by: MP, FACT, DESIG, INSQ, STLE, IFO
	Contains the number of entries -1 in the IL table.	
NLBL	Item	Local Used by: OPTM
	Specifies the number of statement labels encountered while attempting to optimize.	
NLWC	Item	Local Used by: WORST
	Specifies the number of entries -1 in table LWC.	
NO	Input Parameter	Used by: HOLLS
	Specifies the number of characters to be converted.	

NOBDS	Item	Local	Used by: BDSV
	Contains the DICT entry number of the constant 0; however, may contain, temporarily, the DICT entry number of the constant 1 if the 'number of characters' portion of a beaded expression is missing.		
NOPAR	Item	Local	Used by: ANCHR
	Contains a count of the number of parameters encountered in a procedure call.		
NOW	Item	Local	Used by: MP
	Contains the number of working storage assignments currently made in a statement (synonymous with NWS).		
NOW	Item	Local	Used by: WORST
	Specifies the number of working storages -2 assigned prior to the last procedure call.		
NPNL	Item	Local	Used by: WORST
	Specifies the number of entries -1 in table PNL.		
NPWS	Item	Local	Used by: WORST
	Specifies the number of entries -1 in table PWS.		
NSRT	Item	Express	Used by: MP, FACT
	Specifies the number of entries -1 in the SRT table.		
NSS	Item	Express	Used by: ANCHR, SSPRO
	Contains the number of entries in table SS.		
NSS	Input Parameter		Used by: SSPRO
	Specifies the SS table entry associated with the expression.		
NTOI	Procedure		Used by: PRTIL, ERROR, WORST, ILDF
	Obtains the name of a variable described in a DICT table entry from the item IDNS and places it into the item IMAGE.		
NUM	Item	Local	Used by: NTOI
	Specifies the number of characters of an identifier to be transferred to IMAGE.		

10 July 1962

158

TM-555/020/00

NWS

Item

Express Used by: ANCHR, WORST

Specifies the number of temporary storage "registers" currently assigned in a statement.

ODCLS      Switch      Activated in: BDSV  
Sensitive to the class of an entry (CATC).

ODLOC      Item      Local      Used by: ANCHR  
Contains the IL entry number of the last parameter entered into the IL.

ODP      Output Parameter      Used by: POSOD  
Specifies the CAT table entry at which the scan was terminated.

ODPOS      Item      Local      Used by: ANCHR  
Specifies the CAT table entry number of an operand to be entered into the IL table.

OLAY      Subscripted item from table DICT      Express      Used by: DESIG  
If CLAS EQ item, table, array or file:  
    0 = not used in an overlay  
    1 = used in an overlay

OLOC      Item      Express      Used by: MP, FACT, ANCHR  
Contains the IL entry number of the last operator entry generated.

OPFRM      Item      Local      Used by: XFRPT  
Contains the type of logical operator described in the 'anchor point' CAT table entry.  
    1 = OR  
    0 = AND

OPI      Item      Local      Used by: ANCHR  
Indicates whether the last CAT table entry examined described an operator.  
    1 = described an operator  
    0 = described a non-operator

OPLVL      Item      Local      Used by: XFRPT  
Contains the level (or the adjusted level - see description of XFRPT) of the logical operator described in the 'anchor point' CAT table entry.

OPND	Procedure	Used by: MP, DESIG, FACT
	Converts an entry of the CAT table specifying a dictionary referenced variable or a subscript into an IL operand entry.	
OPTM	Procedure	Used by: MP
	Determines the possibility of eliminating GOTO, TEST and RETURN statements, and optimizes transfers implied by such statements by effecting the implied transfer thru the alteration of transfer points contained in IL 'GOTO' entries having generated labels as transfer points or IL 'relational' entries.	
OPTM	Output Parameter	Used by: OPTM
	Indicates whether a GOTO, TEST or RETURN statement can be eliminated.	
	1 = eliminate	
	0 = do not eliminate	
ORIGIN	Subscripted item from table COMMTN Express	Used by: -
	Specifies the starting location for the object program.	
OVER	Table Items OVX	Express Used by: -
	Dummy table.	
OVX	Subscripted item from table OVER	Express Used by: -
	Dummy item.	

P1	Input Parameter		Used by: FACT
	Specifies the CAT table entry number of the starting position of the expression.		
P1	Input Parameter		Used by: OPND
	Specifies the receiving entry number in the IL table.		
P2	Input Parameter		Used by: FACT
	Specifies the operator (LOC, INC, TST) which is to be inserted in the IL.		
P2	Input Parameter		Used by: OPND
	Specifies the entry number of the CAT table to be converted into IL.		
P3	Output Parameter		Used by: FACT
	Indicates the delimiter at the end of the expression. 1 = 'comma' 0 = \$		
P4	Output Parameter		Used by: FACT
	Specifies the terminating CAT table entry of the expression.		
P5	Output Parameter		Used by: FACT
	Error exit used by the procedure.		
PACK	Subscripted item from table DICT	Express	Used by: -
	Specifies the packing of a subscripted item or table. 0 = Null                          3 = Medium (M) 1 = Specified                 4 = Dense (D) 2 = Loose or none (N)		
PAREL	Item	Express	Used by: MP
	Indicates when a FOR statement is part of a parallel group of FOR statements. 1 = part of a parallel group 0 = non-parallel FOR		
PCAT1	Subscripted item from table COMMTN	Express	Used by: -
	Indicates whether a formatted listing of the CAT table entries has been requested of Genl.		



**PCAT2**      Subscripted item from table **COMMTN Express**      Used by: **ANCHR**  
Indicates whether a formatted listing of the CAT table entries has been requested of Gen2.  
1 = formatted listing requested  
0 = formatted listing not requested

**PDAT**      Subscripted item from table **DICT Express**      Used by: **MP, ANCHR**  
If CLAS EQ file:  
0 = fixed length records  
1 = variable  
If CLAS EQ table:  
0 = fixed length  
1 = variable length  
If CLAS EQ item, subscripted item or array and  
TYPV EQ A, D, F, I or K:  
0 = unsigned  
1 = signed  
If CLAS EQ procedure:  
0 = express items are set in the procedure  
1 = no express items are set in the procedure

**PDCTY**      Subscripted item from table **COMMTN Express**      Used by: **-**  
Indicates whether a formatted listing of the DICT table entries has been requested of Gen1.

**PDRQ**      Subscripted item from table **COMMTN Express**      Used by: **MP**  
Indicates whether a formatted listing of the DRQ table entries has been requested of Gen2.

**PI1**      Input Parameter      Used by: **DESIG**  
Specifies the type of expression to be processed.  
0 = GOTO statement  
1 = numeric switch  
2 = item switch

**PI1**      Input Parameter      Used by: **ERROR**  
Specifies the number of the error message.

PI1      Input Parameter      Used by: INSQ  
Specifies the type of statement requiring delayed-processing.  
0 = FOR statement      3 = ORIF statement  
1 = Procedure or Close declaration      4 = IFEITH statement  
2 = IF statement

PI1      Input Parameter      Used by: LNQTH  
Specifies the entry number in the CAT table of the starting position of the scan.

PI1      Input Parameter      Used by: NTOI  
Specifies the DICT entry number of the variable to be processed.

PI1      Input Parameter      Used by: OPTM  
Specifies the label (either by number for generated labels or by DICT entry number for program labels) to be considered in the optimizing process.

PI1      Input Parameter      Used by: PRTHD  
Specifies the heading to be printed.  
0 = IL heading      2 = DRQ heading  
1 = SRT heading      3 = working storage heading

PI1      Input Parameter      Used by: PRTIL  
Specifies the IL entry number of the first entry to be processed.

PI1      Input Parameter      Used by: PRTQ  
Specifies the DRQ entry number of the first entry to be processed.

PI1      Input Parameter      Used by: PRTSR  
Specifies the SRT entry number of the first entry to be processed.

PI1      Input Parameter      Used by: STLE  
Specifies the type of label to be entered into the IL entry.  
1 = generated label  
0 = statement label

PI2      Input Parameter      Used by: DESIG  
Specifies the beginning entry number in the CAT table of the expression to be processed.

PI2      Input Parameter      Used by: INBQ  
Contents depend on the type of entry (PI1).  
    PI1 EQ 0 - specifies the SRT channel number associated with  
            the FOR statement.  
    PI2 EQ 1 - specifies the exit label for the Procedure or Close  
            declaration.  
    PI2 EQ 2,3,4 - specifies the false transfer point associated  
            with the statement.

PI2      Output Parameter      Used by: LNOH  
Specifies the terminal entry in the CAT table of the expression  
scanned.

PI2      Input Parameter      Used by: NTOI  
Specifies the starting character position in item IMAGE into which  
the name is to be placed.

PI2      Input Parameter      Used by: STLB  
Specifies the DICT entry number of the label if PI1 EQ 0 or the  
actual number of the generated label if PI1 EQ 1.

PI2      Input Parameter      Used by: OPIM  
Specifies the type of label contained in PI1.  
    0 = statement label  
    1 = generated label

PI2      Input Parameter      Used by: PRTIL  
Specifies the IL entry number of the last entry to be processed.

PI2      Input Parameter      Used by: PRTQ  
Specifies the DRQ entry number of the last entry to be processed.

PI2      Input Parameter      Used by: PRTSR  
Specifies the SRT entry number of the last entry to be processed.

PI1      Subscripted item from table COMMTN Express      Used by: MP  
Indicates whether a formatted listing of the IL table entries has  
been requested of Gen2.

PLOC	Item	Express	Used by: INSQ
	Contains the DRQ entry number of the last procedure or close.		
PNL	Subscripted item from table PNL	Local	Used by: WORST
	Contains the LWC entry number of the procedure being processed.		
PNLL	Table	Contains item: PNL	Local Used by: WORST
	Used in a push-down manner to regulate the processing of the procedures at a given nesting level (the current entry being used is the nesting level).		
PO1	Output Parameter		Used by: DESIG
	Specifies the terminal entry in the CAT table of the expression processed.		
PO2	Output Parameter		Used by: DESIG
	Specifies the error exit for the procedure.		
POLOC	Item	Local	Used by: ANCHR
	Specifies the IL table entry number of the last assigned temporary storage.		
POS	Item	Express	Used by: ANCHR
	Specifies the CAT table entry currently being examined in the 'level analysis' phase of ANCHR.		
POSOD	Procedure		Used by: ANCHR
	Performs a backward scan thru the CAT table searching for the beginning of an operand to be entered into the IL.		
PPTCT	Procedure		Used by: ANCHR
	Formats and outputs entries from the CAT table.		
PRCD	Item	Express	Used by: MP, WORST PRTIL, ANCHR
	Contains the DICT entry number of the procedure currently being processed. Contains 9000 if the main program is being processed.		

<b>PREN</b>	Item	Local	Used by: BDSV
	Indicates when the left parenthesis of a beaded variable has been encountered.		
	1 = left parenthesis encountered		
	0 = left parenthesis not encountered		
<b>PRI</b>	Item	Local	Used by: SRCHR
	Indicates whether the procedure is scanning inside or outside of a parenthesized expression.		
	1 = inside parenthesis		
	0 = outside parenthesis		
<b>PRNT</b>	Procedure		Used by: PRILL, PRISR, MP, PRTQ, PPTCT, WORST, ERROR
	Outputs the item IMAGE into the file DEBUG.		
<b>PROCN</b>	Item	Express	Used by: MP
	Contains the DICT entry number of the procedure or close currently being processed.		
<b>PRON</b>	Subscripted item from table PWS	Local	Used by: WORST
	Contains the DICT entry number of the procedure or MP (9000) just processed.		
<b>PROS</b>	Subscripted item from table LWC	Local	Used by: WORST
	Contains the DICT entry number of the procedure or MP (9000) just being processed when the call specified in CAWL was made.		
<b>PRTHD</b>	Procedure		Used by: MP, WORST
	Outputs a formatted heading for the IL, SRT and DRQ tables and the starting level of working storage.		
<b>PRILL</b>	Procedure		Used by: MP
	Formats into item IMAGE entries from the IL table.		
<b>PRTQ</b>	Procedure		Used by: MP
	Formats into item IMAGE entries from the DRQ table.		
<b>PRISR</b>	Procedure		Used by: MP
	Formats into item IMAGE entries from the SRT table.		

10 July 1962

167

TM-555/020/00

**PRWS**      Subscripted item from table PWS      Local      Used by: WORST  
Specifies the total amount of temporary storage required by this procedure.

**PSRT**      Subscripted item from table COMMTN Express      Used by: MP  
Indicates whether a formatted listing of the SRT table entries has been requested of Gen2.

**PUTIN**      Procedure      Used by: ANCHR  
Generates IL operand entries from simple operands described in the CAT table; also recognizes complex operands, either subscripted or beaded variables, and directs procedure BDSV to enter these complex operands into the IL.

**PWS**      Table      Contains items: PRON, SLWS, PRWS  
Local      Used by: WORST  
Contains information about the amount of temporary storage and the starting level of temporary storage for the procedures and MP of a program.

10 July 1962

168

TM-555/020/00

QA

Item

Express Used by: MP

Indicates whether an IF or an ORIF statement is being processed.

2 - IF

3 - ORIF

RELAD	Input Parameter	Used by: XFRPT
	Indicates whether the 'anchor-point' is a relational or a logical operator.	
	1 = relational	
	0 = logical	
RIGHT	Item	Express Used by: ANCHR, LVLCP
	Specifies the CAT table entry number of the 'anchor point'.	
RLOC	Input Parameter	Used by: LLOC
	Specifies the starting CAT table entry for the scan.	
RLOC	Item	Local Used by: LVLCP
	Specifies the CAT table entry of a potential 'anchor point'.	
RLOC	Item	Local Used by: XFRPT
	Specifies the CAT table entry that is currently being examined by XFRPT.	
RNGI	Subscripted item from table DICT	Express Used by: -
	If CLAS EQ item, subscripted item or array:	
	0 = no range specified	
	1 = range specified	
RORT	Subscripted item from table DICT	Express Used by: -
	If CLAS EQ item, subscripted item or array:	
	0 = round variable	
	1 = truncate variable	
RRLOC	Input Parameter	Used by: XFRPT
	Specifies the CAT table entry of the last determined 'anchor point'.	
RSLT	Item	Express Used by: ANCHR
	Specifies the CAT table entry of the last operator entered into the IL and where the last temporary storage was assigned.	
RSLTI	Item	Express Used by: ANCHR, PUTIN, BDSV
	Specifies the number +1 of operators which have been processed since the last temporary storage assignment.	



10 July 1962

170

TM-555/020/00

RSRCH      Output Parameter

Used by:    SRCHR

Indicates whether the scan was successful.

0 = unsuccessful

1 = successful

2 = unsuccessful and that the scan was terminated by a \$, or  
else the scan was commenced from within the procedure call.

SAVI      Item      Express      Used by: ANCHR, SSPRO  
Indicates when an ABS modifier has been encountered in a subscript expression.  
1 = ABS encountered  
0 = no ABS encountered

SCKI      Subscripted item from table SS      Express      Used by: ANCHR, SSPRO  
Indicates when a constant has been encountered in a subscript expression.  
1 = found constant  
0 = not found

SEP      Switch      Activated in: ANCHR  
Sensitive to the form (CATF) of an entry with a class of separator (CATC EQ 9).

SEQS      Switch      Activated in: MP  
Sensitive to the form (CATF) of an entry with a class of sequential operator (CATC EQ 8).

SIND      Subscripted item from table DICT      Express      Used by: MP, ANCHR  
If CLAS EQ file:  
0 = Hollerith  
1 = binary  
If CLAS EQ procedure:  
0 = non-function  
1 = function  
If CLAS EQ constant:  
0 = is not initial data  
1 = is initial data  
If CLAS EQ table, item subscripted item or array:  
0 = no initial data  
1 = variable has initial data  
If CLAS EQ statement label or close:  
0 = the label/close was used after its declaration (at least one backward transfer) or else it was used in a switch  
1 = label/close was always used before its declaration (no backward transfer)

**SIZE** Subscripted item from table DICT Express Used by: -  
 If CLAS EQ table and PACK EQ 1:  
     Specifies the number of cards per entry.  
 If CLAS EQ item, subscripted item or array:  
     (1) TYPV EQ H, T:  
         Specifies the number of bytes  
     (2) TYPV EQ A, D, I, S, K:  
         Specifies the number of bits (including the sign bit)  
 If CLAS EQ file:  
     Specifies the maximum number of records  
 If CLAS EQ constant and SIND EQ 1 and the nearest non-constant  
 preceding entry is an array:  
     Specifies the index to the assigned column (relative to 0)  
     for this constant set.

**SLPI** Subscripted item from table SS Express Used by: ANCHR, SSPRO  
 This item indicates when the left parenthesis following an ABS  
 modifier has been encountered in a subscript expression.  
     1 = found left parenthesis  
     0 = not found

**SLWS** Subscripted item from table PWS Local Used by: WORST  
 Specifies the starting level of temporary storage necessary for  
 this procedure.

**SPMI** Subscripted item from table SS Express Used by: ANCHR, SSPRO  
 Indicates when a + or - has been encountered in a subscript  
 expression.  
     1 = found + or -  
     0 = not found

**SPNIL** Item Express Used by: MP  
 Contains the starting IL entry number of a block of IL to be  
 formatted and output.

**SRCHL** Procedure Used by: ANCHR  
 Performs a backward scan thru a statement searching for a logical  
 operator or an IF, IFEITH, or an ORIF sequential operator.

**SRCHR** Procedure Used by: ANCHR  
 Forms a forward scan thru a statement or phase searching for a  
 logical operator.

**SRPI** Subscripted item from table SS Express Used by: ANCHR, SSPRO  
Indicates when the right parenthesis from an absolute value expression has been encountered in a subscript expression.  
1 = found right parenthesis  
0 = not found

**SRT** Table Contains items: SRTA, SRTB, SRTC, SRTD, SRTE, SRTF, SRTG, SRTI, SRTJ, SRTS  
Express Used by: MP, FACT  
Contains information about each FOR statement used in the program.

**SRTA** Subscripted item from table SRT Express Used by: MP  
Indicates if the entry describes at least a 2-factor FOR statement (including FOR ALL type).  
1 = 2-factor or 3-factor FOR  
0 = 1-factor FOR

**SRTB** Subscripted item from table SRT Express Used by: MP  
Specifies the first IL entry number within the range of the FOR statement (specifies the first operator entry following the LOD (ILO EQ 23) operator).

**SRTC** Subscripted item from table SRT Express Used by: MP  
Specifies the last IL entry number in the range of the FOR statement (the entry number of the entry preceding the first operator entry following the TST (ILO EQ 25) entry).

**SRTD** Subscripted item from table SRT Express Used by: MP, FACT  
Specifies the subscript (number) of the subscript activated by the FOR statement ( $1 \leq x \leq 26$ ).

**SRTE** Subscripted variable from table SRT Express Used by: MP  
Contains a count of the level of nesting of FOR statements at the time the FOR statement was encountered.

**SRTF** Subscripted variable from table SRT Express Used by: MP  
Contains the generated label to identify the INC or TST IL operator if a TEST statement has been used in the range of the FOR statement.

**SRTG**      Subscripted item from table SRT      Express      Used by: MP  
Indicates if the FOR statement required a non-dummy TST  
ILO EQ 25) entry.  
1 = non-dummy TST  
0 = dummy TST

**SRTH**      Subscripted item from table SRT      Express      Used by: MP  
Specifies the number of IL entries required to hold the 2nd factor  
of a FOR statement (the INC portion).

**SRTI**      Subscripted item from table SRT      Express      Used by: MP  
Specifies the number of IL entries required to hold the 3rd factor  
of a FOR statement (the TST portion).

**SRTS**      Subscripted item from table SRT      Express      Used by: MP  
Indicates whether the subscript was used as a subscript or as a counter.  
1 = subscript  
0 = counter

**SS**      Table      Express      Used by: ANCHR, SSFRO  
This is a table used in a push-down manner which contains information  
necessary for the analysis of standard subscript expressions.

**SSFRO**      Procedure      Used by: ANCHR  
Recognizes standard subscript expressions in statements.

**SSVI**      Subscripted item from table SS      Express      Used by: ANCHR, SSFRO  
Indicates when a subscripted variable, simple variable or subscript  
has been encountered in a subscript expression.  
1 = found subscripted variable, simple variable or subscript  
0 = not found

**STDI**      Item      Local      Used by: NACHR  
Indicates whether a subscript expression of standard form has  
been encountered.  
1 = subscript expression of standard form  
0 = subscript expression of non-standard form

**STDI**      Output Parameter      Used by: SSFRO  
Indicates whether the expression analyzed was of standard form.  
1 = standard form  
0 = non-standard form

10 July 1962

175

TM-555/020/00

STIB	Item	Express	Used by: PRTIL
	Contains the names of the IL modifier (ILB).		
STIC	Item	Express	Used by: PRTIL
	Contains the names of the "forms" of IL entries (ILC).		
STID	Item	Express	Used by: PRTIL
	Contains the names of the "codes" in IL entries (ILD).		
STIH	Item	Express	Used by: PRTIL
	Contains the names of the IL bead type (ILH).		
STIO	Item	Express	Used by: PRTIL
	Contains the names of the IL operators (ILO).		
STLE	Procedure		Used by: MP, DESIG, ANCHR
	Generates a label IL operator entry (ILO = 16).		
STMC	Item	Express	Used by: MP, INSQ
	Contains a count of the number of statements (not including declarations) processed.		
STOP'IFG	Subscripted item from table COMMEN	Express	Used by: -
	Indicates whether the compilation process is to stop if errors are discovered by the generator.		
STOP'IFT	Subscripted item from table COMMEN	Express	Used by: -
	Indicates whether the compilation process is to stop if errors are discovered by the translator.		
STPC	Item	Express	Used by: MP, INSQ
	A push-down type item used to count BEGINS, START and ENDS, TERM (BEGINS and START increase it by 1, ENDS and TERM decrease it by 1).		
STQA	Item	Express	Used by: PRTQ
	Contains the names of the values of DRQA.		
STSD	Item	Express	Used by: PRTIL
	Contains the names of the subscripts.		

STYPE	Subscripted item from table SS	Express	Used by: ANCHR, SSPRO
	Specifies the type of subscript expression being analyzed.		
	0 = arithmetic		
	1 = bead		
	2 = subscripted variable		
SUBUI	Subscripted item from table ACTS	Express	Used by: MP, ANCHR, SSPRO, BDSV
	Indicates how the subscript was used.		
	1 = used as a subscript (in a subscript expression of standard form) at least once		
	0 = never used as a subscript		
SUBV	Item	Local	Used by: BDSV
	Indicates whether the BDSV is processing within the parenthesis of a beaded variable.		
	1 = operating within the parenthesis		
	0 = not operating within the parenthesis		
SW	Switch		Activated in: DESIG
	Sensitive to the DICT class (CLASS) of a switch point.		
SW1	Switch		Activated In: DESIG
	Sensitive to the DICT class (CLASS) of a GOTO statement.		

10 July 1962

177

TM-555/020/00

T1	Item	Local	Used by: LENGTH
	Push-down type item used to count brackets (stepped by left brackets, decremented by right brackets).		
T1	Item	Express	Used by: MP
	Specifies the number of IL entries required to contain the 2nd and 3rd factor of a FOR statement.		
T2	Item	Express	Used by: MP, DESIG
	Contents depends on use.		
T3	Item	Express	Used by: MP
	Contents depends on use.		
T3	Item	Local	Used by: FACT
	Push-down type item used to count brackets (stepped by left brackets, decremented by right brackets).		
T4	Item	Local	Used by: MP
	Specifies the number of CAT table entries separating the commas in a switch declaration from the next switch point.		
	1 = numeric switch		
	3 = item switch		
T5	Item	Express	Used by: MP
	Contents depends on use.		
TC	Input Parameter		Used by: HOLLS
	Specifies the characters to be converted.		
TFRM	Subscripted item from table DICT	Express	Used by: -
	If CLAS EQ table:		
	0 = null		
	1 = parallel table		
	2 = serial table		
TRAN'ERR	Subscripted item from table COMMTN	Express	Used by: -
	Indicates whether errors were discovered in the object program by the translator.		



TREGN      Subscripted item from table DICT      Express      Used by:    WORST

**If CLAS EQ procedure:**

**Specifies the starting level of internal temporary storage required by the procedure.**

TSS	Item	Local	Used by:	SSPRO
-----	------	-------	----------	-------

**Meaning depends on use.**

TTP	Item	Express	Used by: MP, ANCHR, XRPRT
-----	------	---------	------------------------------

Contains the generated label number of the 'true transfer point' of a relational statement.

**TYPV      Subscribed item from table DICT      Express      Used by: ANCHR**

**Specifies the type of constant, item, subscripted item or array.**

0 = Null	7 = Status (S)
1 = Fixed Point (A)	8 = Octal (O)
2 = Boolean (B)	9 = STC (T)
3 = Dual Fixed Point (D)	10 = Dual Integer
4 = Floating Point (F)	11 = Dual Octal (Q)
5 = Hollerith (H)	12 = (Error)
6 = Integer (I)	

10 July 1962

179

**TM-555/020/00**

WCNT	Item	Local	Used by: WORST
	Contains the 'running' level of working storage required by the procedures processed.		

WHERE	Item	Local	Used by:	ANCHR
	Indicates whether a Boolean operand requires a level.			
	1	requires a level		
	0	no level required		

WORST	Procedure	Used by: MP, ANCHR
	Optimizes the amount of temporary storage required by the procedures of a program.	

XFRPT	Procedure	Used by: ANCHR
	Determines the linkage (via transfer points) necessary in expressions involving relational and logical operators and adjusts GTO operators and generates labels to effect the necessary linkage.	
XXX	Output Parameter	Used by: ANCHR
	Specifies the error exit for the procedure.	
XXX	Output Parameter	Used by: PUTIN
	Specifies the error exit for the procedure.	
XXX	Output Parameter	Used by: SRCHL
	Specifies the alternate exit for the procedure.	
XXX	Output Parameter	Used by: SRCHR
	Specifies the error exit for the procedure.	
XXX	Output Parameter	Used by: SFRPT
	Specifies the error exit for the procedure.	

### C. DESCRIPTION OF THE GEN2 MAIN PROGRAM (MP) REGIONS

#### INTRODUCTION

The main body of Gen 2 is composed of regions, each of which is identified with a particular statement type. The type of a statement is determined from the class (CATC) of the first entry in the CAT table describing the statement. (This analysis is accomplished by activating the switch CLASS.) Some classes are further broken down by their form (CATF). (This, too, is accomplished via switches, SEQs and BRAKS.) The regions of the program are labeled according to the class and perhaps the form of the statement type with which they are associated. Thus, the 'LO1' region processes statement labels (class of 1); the 'LO8\_\_' region is associated with sequential operators (class of 8) which are further broken down by forms: LO80 pertaining to IF statements (form of 0), LO81 pertaining to IFEITH statements (form of 1) and LO82 pertaining to ORIF statements (form of 2). The delayed-processing mentioned above, is performed in the end-of-statement, ENDST, region. This region is entered following the processing of each statement to determine if delayed-processing is needed and to perform it if required. (Statements containing errors enter this region via an alternate path which first locates the end (\$) of the statement.) The actual delayed-processing region is broken into sub-regions according to the types of delayed-processing. Thus, the LNO region is associated with a DRW entry of type 0 (FOR), LNL with a DRQ entry type 1 (procedure and close) etc.

1. a. Name - L00  
b. Description - CAT table entries to be processed are input in this region.  
d. Operations -
  1. Input CAT table entries.
  2. Delay until the input operation is complete.
  3. If no entries could be input (end-of-file on input tape).
    1. Set DONE = 1 to indicate finished processing (at L107A).
    2. Exit to output the IL (to WRIL).
  4. Exit to analyse the type of statements to be processed (to NEXT1).
2. a. Name - ~~NE~~<sup>X</sup>ST1  
b. Description - Empties the temporary push-down queue when it is full, determines the type of statement to be processed and directs the processing thereof.  
d. Operations
  1. Set LOC=NIL+1 to remember the beginning IL entry for the next statement.
  - (NEXT) 2. Increment CAT table control subscript to specify the first entry of the next statement (C=C+1).
  3. If the temporary push-down queue is not empty and the next entry in the CAT table does not describe an END (see description of DRQ table, DNRQ and INSQ).
    1. Transfer the entries from the temporary push-down queue into the DRQ table.
    2. If the last entry in the DRQ table is associated with a procedure (DRQA EQ 1)
      1. Set PLOC=NRQ to remember the entry.
  4. Activate the switch CLASS to analyse the type of statement to be processed and to direct the processing thereof.
3. a. Name - L01  
b. Description - This region generates a label IL operator entry for a statement label.  
c. Procedures called - STLE  
d. Operations -
  1. Enter the statement label into the IL table using procedure STLE.

3. d. 2. Reset the item LPLUS ( a new count is started with each new label).
  3. Set item LNAME to specify the DICT entry number of the label just encountered (CATF).
  4. Exit (since a label is encountered only at the beginning of statements, this region exits to a point at the end of the end-of-statement) (to END 2).
4. a. Name - L02
  - b. Description - Assignment statements and procedure call statements are processed in this region.
  - c. Procedure called - ANCHR, ERROR
  - d. Operations -
    1. Set the boundary entries in the CAT table of the statement for ANCHR (LAST and FIRST).
    2. Set FTP = 0 (to be later used as a test for an illegal operator)
    3. Process the statement using ANCHR.
    4. If a relational operator was encountered by ANCHR (FTP NQ 0)
      1. Output error message #14
      2. Exit to locate the end of the statement
    5. If the statement processed was a Boolean assignment statement (BV NQ 0)
      1. Generate IL information entries specifying BV as both boundaries (for possible optimizing of the GOTO inserted by ANCHR)
    6. If not processing the main part of a program (PRCD NQ 9000)
      1. If the statement processed was an assignment or exchange statement (ILO EQ 21,22)
        1. If the first operand of the IL assignment or exchange statement was an express item (DEFN EQ 1)
          1. Set PDAT to 0 (indicating that an express item was set in a procedure)
        2. If the second operand of the IL exchange statement was an express item (DEFN EQ 1)
          1. Set PDAT to 0 (indicating that an express item was set in a procedure)
    7. Exit

5. a. Name - ERR
  - b. Description - This region outputs an error message for statements commencing with illegal parts of speech.
  - c. Procedures called - ERROR
  - d. Operations -
    1. Output error message #1
    2. Exit to find the end of the statement
6. a. Name - L083
  - b. Description - GOTO statements are processed in this region.
  - c. Procedures called - OPTM, LNTH, DESIG
  - d. Operations -
    1. If the statement is erroneous, print error message and then exit.
    2. Set SIND of the label associated with the GOTO to 0 indicating a potential backward transfer is being made to the label by the GOTO (if SIND is still set at the end of the program, the last transfer to the label was a backward transfer)
    3. If a procedure is being processed (PRCD NQ 9000) and the associated label is an express statement label (DEFN = 0, CLAS = 1)
      1. Set PDAT = 0 meaning express label transferred to from within a procedure
    4. If the associated label is a statement or program label (CLAS EQ 1 or 12)
      1. Attempt to optimize away the GOTO with procedure OPTM
        1. If successful
          1. Skip the GOTO
          2. Exit
    5. If the associated label identifies a switch which is subscripted
      1. Determine the length of the expression using procedure LNTH (saving the length in the CAT table (CATF))
    6. Process the GOTO using procedure DESIG
    7. Exit

7. a. Name - L081  
b. Description - IFEITH statements are processed in this region.  
c. Procedures called - INSQ  
d. Operations -
  1. Make an entry in the DRQ table using INSQ so that the delayed-processing required by the IFEITH will be performed
  2. Generate a label to identify the bottom of the IFEITH statement (IFEIX = GLAB)
  3. Consider the 'IFEITH' as a 'BEGIN' thereby stepping the statement parenthesis counter (STPC)
  4. Continue as if the entry were an ORIF (L082)
8. a. Name - L082  
b. Description - ORIF statements are processed in this region.  
d. Operations -
  1. If the statement has the structure ORIF 1
    1. Eliminate the statement
    2. Exit
  2. Set item QA to 3 (indicating ORIF)
  3. Proceed as if an IF statement (except for the setting of QA) (L080)
9. a. Name - L080  
b. Description - IF statements are processed in this region.  
c. Procedures called - ANCHR, ERROR, STLE, IFO  
d. Operations -
  1. Set item QA to 2 (indicating IF)
  2. Set the boundary entries in the CAT table of the statement for ANCHR (LAST and FIRST)
  3. Clear FTP (to be later used as a test for an illegal operator)
  4. Process the statement using ANCHR
  5. If an '=' operator was encountered by ANCHR(FTP EQ 0)
    1. Output error message #15
    2. Exit to locate the end of the statement



9. d. 6. If the statement contained 'OR's (TTP NQ 0)
  1. Enter the true transfer point label for the statement in the IL using STLE
  2. Generate information entries associated with the label using procedure IFO with it obtaining its information from the temporary push-down queue.
  3. Set the lower information boundary to the last entry of the IL entry used (ILI = NIL)
7. Make an entry in the DRQ table (so that the delayed-processing required by the statement will be performed)
8. Exit

10. a. Name - L084

b. Description - FOR statements are processed in this region.

c. Procedure called - FACT, LENGTH, ERROR, INSQ

d. Operations -

1. General Philosophy -

The type of FOR statement being processed is determined from context. Depending on the type of 'FOR' statement and its use or non-use in a parallel FOR manner, the program recognizes which of the IL operators (LOD, INC, TST) are required by the statement. For simple 'FOR' statements, FOR I = VAR \$, LOD and dummy TST operators are required. For 2-factor 'FOR' statements, FOR I = VAR, VAR, VAR \$, or FOR ALL statements, FOR I = ALL (TAB) \$, LOD, INC, and TST operators are required. The factor itself is analyzed by procedure FACT. In the event that a factor contains a complex arithmetic expression, FACT directs ANCHR to process the expression. If the factor is simple, OPND is used to put the factor into IL. In either case FACT adds the appropriate FOR operator (LOD, INC, TST) to the IL expression. If parallel FOR statements are encountered, it is necessary to determine which is the controlling FOR statement of the group (the first FOR ALL statement or the first statement containing 2 or 3 factors).

The entire FOR statement is expanded into IL at the same time. However, it is necessary that the INC and TST expressions eventually be relocated at the bottom of the loop (after the statement succeeding the FOR statement or the last FOR statement in a parallel group). The relocation takes place as the delayed-processing of a FOR statement. At this time all of the entries associated with the INC and TST are moved to the bottom of the loop and their original locations are cleared. (Also, at this time, the subscript is deactivated and the label identifying the top of the loop is entered into the IL after the LOD).

More information is required by the Translators for FOR statements than can be contained in the IL entries. Included in this category is information like the number of IL entries required to hold the INC and TST expressions, the first and last IL entry in the range of the FOR statement and the nesting level of FOR statements. As FOR statements are processed this information is accumulated in entries of the SRT table (one entry is associated with each FOR statement) and this table is passed along to the Translators with the IL table.

10. d. 2. General Operations -

1. If the subscript referenced in the FOR statement is already active (ACTSS EQ 1)
  1. Output error #2
  2. Exit to locate the end of the statement
2. If the type of FOR statement is of the FOR ALL variety (CATB EQ 7)
  1. Set BOOL = 1
3. Use procedure FACT to analyze and expand the first factor (the LOD expression)
4. Activate the subscript (ACTSS = 1)
5. If the statement has 2 factors (BOOL1 EQ 1 meaning FACT found a comma as the delimiter of the factor) or is a FOR ALL (BOOL EQ 1)
  1. Use procedure FACT to analyze and expand the second factor (the INC expression)
6. If the statement is in a group of parallel FOR statements, i.e., the last entry in the DRQ table describes a FOR statement (DRQA EQ 0) and there have been no intervening statements (simple or compound) (DRQA EQ STPC and DRQC + 1 EQ STMC)
  1. Update the entry in the DRQ table to reflect the new FOR (set DRQC = STMC, step FOR counter DRQE)
  2. If this is not the controlling FOR statement of the group (DRQF EQ 1) and it does have a 3rd factor (SRTA EQ 1)
    1. Ignore the 3rd factor (skip it using LENGTH)
  3. If it is the controlling 'FOR'
    1. Proceed at d.2.8.1
7. If the statement is not in a parallel group
  1. Make an entry in the DRQ table (to effect the delayed-processing required)

10. d. 2. 8. If the statement has a 3rd factor (SRTE EQ 1)
    1. Use procedure FACT to analyze and expand the factor (the TST expression)
    2. Set SRTE to 1 (non-dummy TST inserted)
    3. If FACT terminated on a comma (BOOL3 EQ 1)
      1. Output error #3
    9. If the statement requires a "dummy TST" (a 1-factor or a non-controlling 2- or 3-factor FOR)
      1. Set up a 'dummy TST' in IL (contains an operand entry specifying DICT entry 0 and ILY set to 1)
    10. If a label is required to identify the top of the loop (necessary for 2- or 3-factor FOR (BOOL1 EQ 1 and DRQF NQ 1))
      1. Generate a label (GLAB = GLAB+1)
      2. Save the label for the delayed-processing
        1. If the FOR is non-parallel save the label in DRQI[DNRQ]
        2. If the FOR is parallel, save the label in DRQI[DNRQ]
      3. Set DRQF = 1 (indicates controlling FOR)
      4. Set the label into the transfer point of the TST operator (ILE = GLAB)
    11. Set SRTE to the current nesting level (SRTE = NEST)
    12. Exit
  3. Error return from procedure FACT
    1. Deactivate the subscript (ACTSS = 0)
    2. Output error message #18
    3. Exit to locate the end of the statement
11. a. Name - L086
  - b. Description - STOP statements are processed in this region.
  - d. Operations -
    1. Generate an IL stop (ILO = 20) entry
    2. If there is a label included in the STOP statement
      1. Put the label into ILF
    3. Exit

12. a. Name - L085
  - b. Description - RETURN statements are processed in this region.
  - c. Procedures called - OPTM
  - d. Operations -
    1. If it is possible to eliminate the RETURN using procedure OPTM
      1. Exit
    2. Generate an IL GOTO (ILO = 15) entry with an operand describing the terminating label of the procedure or close currently being processed (the label was saved in DRQI at the entry specified in PLOC)
    3. Exit
- 
13. a. Name - L087
  - b. Description - the TEST statement is processed in this region.
  - c. Procedures called - OPTM, ERROR
  - d. Operations -
    1. If there is a subscript included in the statement (example: TEST Q \$)
      1. Set B1 = 1
    2. If the statement is not in the range of any FOR statement (NDRQ EQ 0, meaning no previous statements require delayed-processing and therefore, in particular, no FOR statements)
      1. Output error #5
      2. Exit to locate the end of the statement.
    3. Scan the DRQ table for entries describing FOR statements (DRQA EQ 0)
      1. If a FOR entry is encountered
        1. Set B2 = 1 (meaning the TEST statement is in the range of some FOR statement)
        2. Scan the entries of the SRT table referenced in the DRQ entry (DRQD specifies entry number, DRQE specifies number of entries)
        3. If an entry is encountered referencing at least a two-factor FOR statement (SRTA EQ 1 or SRTN NQ 0)
          1. If a directed TEST is being processed (B1 EQ 1)
            1. If the subscript referenced in the TEST does not match the subscript referenced in the SRT entry
              1. Continue the scan d.3.1.2

13. d. 3. 1. 3. 2. If the SRT entry does not include a transfer point for a TEST transfer (SRTF EQ 0)
    1. Generate a label and put it into the SRT entry (SRTF)
    3. If it is possible to eliminate the TEST using procedure OPTM
      1. Exit to locate the end of the statement
    4. Generate an IL GOTO entry (ILO = 15) specifying in the operand entry the TEST transfer label (SRTF)
    5. Exit to locate the end of the statement
  4. If this is a directed TEST and its subscript matches that referenced in the SRT entry
    1. Output error #6 (TESTing single factor FOR)
    2. Exit to locate the end of the statement
  4. At this point no adequate FOR statement was found. If at least one DRQ FOR entry was encountered (B2 EQ 1)
    1. If a directed TEST was being processed
      1. Output error #8
      2. Exit to locate the end of the statement
    2. Output error #7
    3. Exit to locate the end of the statement
  5. Output error #5 (TEST used outside range of a FOR statement)
  6. Exit to locate the end of the statement
14. a. Name - L09
  - b. Description - Separators (CATC EQ 9) are processed in this region.
  - c. Procedures called - ERROR
  - d. Operations -
    1. If the separator is not a dollar sign (CATF EQ 4)
      1. Output error #13
      2. Exit to locate a dollar sign
    2. Exit

- 15. a. Name - L100
  - b. Description - Statements commencing with illegal brackets are processed in this region.
  - c. Procedures called - ERROR
  - d. Operations -
    - 1. Output error #21
    - 2. Exit to locate the end of the statement
- 16. a. Name - L104
  - b. Description - Bracket `BEGIN` is processed in this region.
  - d. Operations -
    - 1. Step push-down counter `STPC`
    - 2. Exit (do not do delayed-processing after this bracket)
- 17. a. Name - L105
  - b. Description - Bracket `END` is processed in this region.
  - c. Procedures called - ERROR
  - d. Operations -
    - 1. Decrement push-down counter `STPC`
    - 2. If push-down counter `STPC` is 0 (meaning too many `ENDs` in the program), set `STPC = 1` and output error 17.
    - 3. Adjust the statement parenthesis count in the temporary push-down queue to the latest value of the statement parenthesis counter (`DRQB[1] = STPC`).
    - 4. Exit
- 18. a. Former name - L106
  - b. This region has been eliminated. `START` is ignored. All initialization is done at the beginning of the phase

19. a. Name - L107
- b. Description - Bracketer TERM is processed in this region.
- c. Procedures called - ERROR
- d. Operations -
  1. Decrement push-down counter STPC
  2. If push-down counter STPC is not 0 (too many BEGINS in program)
    1. Output error #9
  3. Generate an IL term entry (ILO = 19)
  4. If a label is included in the statement
    1. Add the label to the term IL entry
  5. Set DONE = 1
  6. Exit to terminate all processing (see region L14.d.3.1.6)
20. a. Name - L14
- b. Description - Switch, Procedure and Close declarations are processed in this region as well as the terminating operations performed at the end of the program (outputting the IL and SRT tables, etc.)
- c. Procedures called - DESIG, LENGTH, PRTHD, PRTIL, PRTSR, PRTQ, BTOD, PRNT, ERROR, INSQ, WORST
- d. Operations -
  1. For SWITCH declarations (CATF EQ 0)
    1. General Philosophy -

Switch declarations are processed one switch-point at a time. Switch-points specifying statement or compool program labels are entered directly into the IL table as operands of the switch. However, for each switch-point specifying a switch or close label, it is necessary to substitute a generated label and to effect the actual transfer to the switch or close as if the point were a GOTO statement located at the bottom of the switch. Therefore, it is necessary to process the declarations twice. The first time the non-switch and non-close points are entered into the IL and generated labels are substituted for the switch and close points. The second time the declaration is processed, only the switch and close-points are considered and they are processed like GOTO statements located after the declaration and identified by their appropriate generated label. Because there are these internal and external parts of a switch, it is necessary that the boundaries of the switch be marked in the IL. This is accomplished by repeating the IL entry containing the switch

name at the top and at the bottom of the switch (below any GOTO switch entries associated with the declaration). Example:

JOVIAL: SWITCH SW = (SLAB<sub>1</sub>, SW<sub>2</sub>, SLAB<sub>2</sub>) \$

```
IL      : SWN SW
          SLAB1 (statement label)
          GLABx (generated label)
          SLAB2 (statement label)
          LBL GLABx
          GOTO
          SW2 (switch name)
          SWN SW (duplicate of top of switch)
```

20. d. 1. 2. General Operations -

1. Generate a Switch Name IL operator entry (ILO = 17) including the name of the switch
2. For an 'item' switch declaration
  1. Set T3 = 2 to indicate item switch
  2. Set T4 = 3 to indicate the number of entries to be skipped between points
  3. Continue at d.1.2.4
3. For a 'numeric' switch declaration
  1. Set T3 = 1 (numeric switch)
  2. Set T4 = 1
4. Set T5 to the beginning CAT table entry of the switch (to be used to indicate where to start the second time thru)
5. Process the switch points
  1. If the constant associated with a point from a numeric switch is negative
    1. Set MYNUS = 1
  2. Process the point using procedure DESIG
  3. If the processing is not complete for the declaration "continue at  d.1.2.5."
  4. If this was the first time thru the declaration (T3 not set to GOTO)
    1. Set T3 to 0 (indicating processing a GOTO statement)



20. d. 1. 2. 5. 4. 2. Set the processing to begin at the entry specified in T5
  3. Continue at d.1.2.5 processing the points as GOTO statements
  6. Duplicate the beginning of the switch at the bottom
  7. Exit
2. For declarations other than Switch, Procedure or Close
  1. Output error #12 (illegal declaration)
  2. Exit to find the end of the declaration
3. For Procedure or Close declarations
  1. For a Procedure declaration
    1. Set PDAT = 1 (will remain set if no express items are set in the procedure)
    2. Ignore the parameters in the procedure (using procedure LENGTH to skip by them)
    3. If this is the first procedure of the program (PRCD EQ 9000)
      1. Operate procedure WORST for the main part of the program (non-procedure)
    4. Output the binary IL table
    5. If a formatted listing of the IL table is requested (PIL EQ 1)
      1. Output the heading using PRTHD
      2. Output the table using PRTIL
    6. If this is the last procedure of the program or a TERM statement has been encountered (DONE EQ 1)
      1. Output the SRT table if requested (binary or formatted)
      2. Output the DRQ table (formatted) if requested
      3. Optimize the working storage using procedure WORST
      4. Output the number of statements in the program (STMC)
      5. Shut the files
      6. Exit
    7. Clear the IL table
    8. If the statement parenthesis counter (STPC) does not equal 1 (indicates too few ENDS in the preceding procedure or main program)
      1. Output error # 20

20. d. 3. 1. 8. 2. Set STPC = 1
  2. For a Close declaration
    1. Set SIND to 1 (if it remains set until the end of the program there have been only forward transfers to the Close)
  3. Make up Procedure/Close entrance IL operator entry (ILO = 13)
  4. Generate a label to identify the end of the procedure (GLAB = GLAB+1)
    1. Put the label in the entrance entry (ILE = GLAB)
  5. Make an entry in the DRQ table using procedure INSQ (so that the delayed-processing for the procedure will be performed, i.e., putting in the exit of the procedure)
  6. Exit
21. a. Name - L15
  - b. Description - Input-Output statements are processed in this region
  - c. Procedures called - OPND, ANCHR
  - d. Operations -
    1. If a receptacle is included in the statement (example: OPEN INPUT FILE'NAME VAR, VAR is receptacle)
      1. For simple receptacle (not subscripted)
        1. Enter the receptacle into the IL table using procedure OPND
      2. For complex receptacles
        1. Use procedure ANCHR
      3. If the receptacle is express (DEFN EQ 1), a procedure is being processed (PRCD NQ 9000) and the type of the statement is 'input'
        1. Set PDAT = 1 (express item being set)
    2. Generate an I/O IL operator entry (ILO = 28,29,30,31,32,33)
    3. Exit
22. a. Name - L17
  - b. Description - Direct code statements are processed in this region
  - d. Operations -
    1. Generate a direct code IL entry (ILO = 27, ILF = CATF 1st card #, ILE = CATE number of cards)
    2. Exit

23. a. Name - SEMI
- b. Description - Searches for the dollar sign ending a statement are performed in this region
- d. Operations -
1. Scans forward looking for an entry describing a dollar sign (CATC EQ 9, CATF EQ 4)
  2. Exit for end-of-statement and delayed-processing
24. a. Name - ENDST
- b. Description - This is the end-of-statement region where certain book-keeping functions required after each statement are performed and from where the delayed-processing region is entered.
- d. Operations -
1. If the statement just processed was not a direct code statement
    1. Place the count of the number of statements since the last statement label (LPLUS) into the first IL operator entry (specified in LOC) of the statement (ILE = LPLUS)
    2. Step the count of the statements since the last label (LPLUS = LPLUS+1)
    3. Step the statement counter (STMC)
    4. Exit for delayed-processing
25. a. Name - ENDL
- b. Description - This is the control region for delayed-processing
- c. Procedures called - PRTHD, PRTQ
- d. Operations -
1. If there are no entries in the DRQ table (NDRQ EQ 0 meaning no possible delayed-processing)
    1. Exit to process the next statement
  2. If one simple or compound statement has been processed since the last entry was made in the DRQ table (STPC LS DRQB and STMC-DRQC GR 1 meaning there is delayed-processing to be performed)
    1. If a formatted listing of the DRQ entries is requested (PDRQ EQ 1)
      1. Output the entry using procedures PRTHD and PRTQ
    2. Activate the switch ENDS to process the DRQ entry
  3. Exit to process a new statement

## 26. a. Name - LNO

- b. Description - The delayed-processing required by a FOR statement is performed in this region
- c. Procedures called - STLE
- d. Operations -

Note: The delayed-processing required for all of the FOR statements contained in a parallel group is performed at one time. The starting SRT entry and number of FOR statements in the group are specified in the DRQ entry (DRQD and DRQE). For a single FOR statement the number of FOR statements specified in DRQE is 1.

- 1. Set SRTS = 1 if the subscript was ever used as an index (SRTS = SUBUI)
- 2. Deactivate the subscript
- 3. If the FOR statement has a dummy TST entry (SRTG EQ 0) and an information entry was just made in the IL (ILO EQ 34)
  - 1. Do not erase the information (NIL = NIL+2)
- 4. If the FOR statement was referenced by a TEST statement (SRTF NQ 0)
  - 1. Put the required label into the IL using procedure STLE
- 5. Relocate the IL entries comprising the INC and TST expressions to the next available entries in the IL and clear their old locations
- 6. Reduce the count of the nesting of FOR statements (NEST = NEST -1)
- 7. If there is a label associated with the top of the FOR loop (DRQI NQ 0)
  - 1. Enter the label into the IL at the entry following the LOD expression
- 8. Exit

## 27. a. Name - LNI

- b. Description - The delayed-processing required by Close and Procedure declarations is performed in this region
- c. Procedures called - STLE, WORST
- d. Operations -
  - 1. If the last entries made in the IL table describe a RETURN statement (an IL GOTO entry with operand specifying the exit label of the Procedure/Close)
    - 1. Eliminate the statement by clearing the IL entries
  - 2. Enter the exit label for the Procedure or Close into the IL table using procedure STLE

27. d. 3. Generate an IL return entry (ILO = 14) for the Procedure or Close  
4. If this is the delayed-processing for a procedure (CLAS EQ 6)  
1. Record the temporary working storage required by the procedure using WORST  
5. Exit
28. a. Name - LN2  
b. Description - The delayed-processing required by an IF statement is performed in this region  
c. Procedures called - STLE, IFO  
d. Operations -  
1. Enter the false transfer point of the IF (or ORIF) statement into the IL using procedure STLE  
2. Enter some information for the label with procedure IFO (this information specifies the first and last entry number of the section of IL in which the label is used)  
3. Exit
29. a. Name - LN3  
b. Description - The delayed-processing required by an ORIF statement is performed in this region  
c. Procedure called - OPTM  
d. Operations -  
Note: The delayed-processing for an ORIF statement entails the generation of the implied GOTO statement (to the end of the IFEITH statement) inherent in the definition of the IFEITH-ORIF statement as well as the insertion of the false transfer point of the ORIF.  
1. Try to eliminate the implied GOTO using procedure OPTM  
1. If successful continue at region LN2  
2. If the last entries made in the IL describe a relational statement (meaning a previous GOTO has been optimized out)  
1. Proceed at LN2  
3. Generate a GOTO IL operator entry (ILO = 15) with an operand specifying the end of the IFEITH statement (IFEIX)  
4. Proceed at LN2

30. a. Name - LN4
- b. Description - The delayed-processing required by the IFEITH statement is performed in this region.
- c. Procedures called - STLE
- d. Operations -
1. Enter the terminating label of the IFEITH statement (IFEIX) using procedure STLE
  2. Enter information entries for the label using procedure IFO
  3. Perform a backward scan thru the IL searching for the implied GOTO statement for the last ORIF portion of the IFEITH statement
    1. If the scan is successful eliminate the GOTO statement
  4. If this is a nested IFEITH statement (DRQD NQ 0)
    1. Set the terminating label of the next IFEITH into IFEIX (IFEIX = DRQD)
  5. Exit
31. a. Name - END2
- b. Description - Determines if additional CAT table entries are needed for processing
- d. Operations -
1. If the last CAT table entry has been processed
    1. Exit to input more entries
  2. Exit to process the next statement

10 July 1962

201

TM-555/020/00

**D. DESCRIPTION OF THE GEN2 PROCEDURES**

## 1. a. Name - ANCHR

b. Description: This procedure analyzes relational and arithmetic expressions to determine the order in which the expression must be performed and generates IL entries representing the expression (in the newly determined order).

## c. Parameters

## 2. Output

1. XXX - Error exit from the procedure

## d. Local Items

1. OPI	7. WHERE	13. BDIND	19. NOPAR
2. LN	8. STDI	14. ERAS	20. ODLOC
3. COMI	9. LBLOC	15. ILLOC	
4. BASE	10. ASNTP	16. POLOC	
5. BDCNT	11. ODPOS	17. AUI	
6. CCNT	12. LEVEL	18. IOPI	

## e. Express Items

1. BNWS	8. RSLTI	15. TTP	22. IL table items
2. RIGHT	9. LBGR	16. FTP	23. DICT table items
3. LEFT	10. POS	17. LBGR	24. LOGIC table items
4. RSLT	11. ATTP	18. ODLOC	25. COMMA table items
5. MID	12. AFTP	19. BV	26. BEADS table items
6. ASPAC	13. CALL	20. NOW	27. ACTSS table items
7. NWS	14. NIL	21. CAT table items	28. SS table items
			29. NSS table items
			30. MAXWS

## f. Procedures Called

1. SRCHL	5. PUTIN	9. BDSV
2. SRCHR	6. ERROR	10. XFRPT
3. SSPRO	7. POSOD	11. PPTCT
4. LVLCP	8. LLOC	12. STLE

## g. Used by:

1. MP  
2. DESIG  
3. FACT



## 1. h. Operations

Note: See SP-127 for a discussion of the algorithm used as a basis for this procedure.

### 1. General Philosophy

The procedure is divided into two main sections. The first section is called the level assignment phase and the second section is the level analysis phase. The level assignment phase consists of a scan thru a statement assigning levels or priorities to the various operators encountered. During the first scan a 'push-down' type counter (BASE) is used to recognize when operators are encountered within bracketed expressions. The priorities or levels are raised by a fixed amount (the value of BASE) to effect the nesting or ordering implied by the parenthesis and brackets. Once the levels have been assigned, an analysis of the levels is made to determine the actual ordering of the expression by the 'level analysis' phase. This phase makes a second pass thru the statement and determines the ordering by the following rule: if the priority or level assigned to an operator is less than or equal to the level assigned to the operator immediately preceding or to the left of it, then the operator on the left and its operands should be entered into the IL table (the operator on the right on this case is called an 'anchor-point').

### 2. 'Level Assignment'

#### 1. General Philosophy

This section makes a scan through a statement assigning levels or priorities to the operators it encounters. The values of the priorities are presented below:

Operator	Priority
1. Separators ('.', ',', '=', '==', '...')	1
2. Logical operator 'OR'	2
3. Logical operator 'AND'	3
4. Relational operators ('LS', 'EQ', 'LQ', 'GR', 'NQ', 'GQ') and Boolean items requiring testing (example IF BOOL \$ as opposed to BOOL1 = BOOL2 \$)	4
5. Arithmetic operator '+', '-'	5

6.	Arithmetic operator '*', '/'	6
7.	Arithmetic operator '**'	7
8.	Unary minus	8
9.	Procedure call ('OF')	10
10.	Absolute value (ABS)	11
11.	Nested subscript of standard form	12

(See further on in the discussion of this procedure for a definition of standard form.)

The level of nesting of expressions is maintained in a 'push-down' type item called BASE. This item is increased by 20 for every left parenthesis or bracket and decreased by 20 for every right parenthesis or bracket. The only exceptions to this rule are the brackets around standard subscript expressions and the parenthesis around a beaded variable. These types of bracketers do not alter the value in BASE. Therefore, the actual level assigned to a given operator is a function of its priority and the current base value.

LEVEL = 'Priority' + BASE

#### 1. h. 2. 2. Operations

1. Housekeep key items
2. Commence scan at entry specified in item FIRST
3. When the scan proceeds to the entry specified in LAST
  1. If a formatted listing of the CAT table thus processed is requested (PCAT2 EQ 1)
    1. Format and output the CAT table using procedure PPTCT
    2. Continue processing in the analysis section 1.h.3.
4. Process each entry by its class (CATC) by activating the switch CLSS
  1. Separator processing according to the form (CATF) of the entry by activating the switch SEP
    1. For '.', '=', '=='
    1. Assign level to operator

1. h. 2. 2. 4. 1. 1. 2. Set the 'operator just processed' indicator (OPI = 1)

2. For ',', or '...'

Note: It is necessary to be able to distinguish between commas ending subscript expressions and commas ending procedure call parameter expressions. The reason for this is that commas ending subscript expressions must be processed like subscript brackets because of the analysis done in the recognition of subscript expressions of standard form (see discussion in section on left brackets). This recognition is accomplished using the 'push-down' table COMMA. A 1 valued entry is made in this table each time a left bracket is encountered and a 0 valued entry is made for each left parenthesis. Right brackets and right parenthesis delete entries. Therefore, by examining the current entry, it is possible to determine if the comma is located within subscript brackets or within parenthesis.

1. If this separator terminates a bead or subscript expression (CIND EQ 1 meaning the separator was encountered within a bracketed expression)

1. Set COMI = 1

2. Continue processing as if the separator were a right bracket (h.2.2.4.10.4)

3. For '\$'

1. Set LAST to the CAT entry number of the preceding entry (LAST = POS - 1), thus terminating the scan.

2. Continue processing at h.2.2.3

2. Arithmetic operator processing according to the form (CATF) of the entry by activating the switch AOP

1. For '+', '-', Unary -, ABS

1. Assign level

1. h. 2. 2. 4. 2. 1. 2. Set OPI = 1
2. For '\*', '/', '\*\*'
  1. If the preceding entry contained an operator (OPI EQ 1)
    1. Output error #26
    2. Continue scan (h.2.2.4)
  2. Assign level
  3. Set OPI = 1
3. For procedure call 'OF'
  1. Assign base
  2. If a 'NOT' has been encountered previously in the statement (LN is odd)
    1. Exclude from the influence of the 'NOT' any Boolean expressions which are included as parameters in the procedure call (step LN by 1 and set LOGIC to BASE + 20)
  3. Set OPI = 1
3. Logical operator processing according to the form (CATF) of the entry by activating the switch LOP
  1. For a 'NOT'

Note: 'NOT' symbols themselves are eliminated from the statements. However, their influence is effected in the statement according to the following rules:

1. Boolean operands are to be negated;
2. Relational operators are to be complemented;
3. Logical expressions are to be altered in accordance with De Morgan's theorems:

$$\begin{aligned} \text{(a)} \quad \text{NOT}(A \wedge B) &= \text{NOT } A \vee \text{NOT } B \\ \text{(b)} \quad \text{NOT}(A \vee B) &= \text{NOT } A \wedge \text{NOT } B \end{aligned}$$

The push-down table LOGIC is used to acknowledge the presence of a NOT influence. An entry is made in the table each time a NOT is encountered. The base level at which the NOT is to be effective is recorded in the entry. This is done even when the NOT is to affect only one relational operator or Boolean item where the base level will not increase. However, the system continues to work because the condition for deletion of an entry is met (see below).

Example: (NOT A LS B)<sub>1</sub>

NOT (A LS B LS C)<sub>2</sub>

In expression 1 the base at the NOT and at the LS is the same. However, in expression 2 the base has been raised for the LS. An entry is deleted from the table each time the base drops below the value recorded in the entry. The test for deleting an entry is made after the processing of relational operators, Boolean expressions and right bracketers. If the number of entries in this table is odd, an operator is under the influence of a NOT. Thus in effect even numbers of nested NOT's will tend to cancel themselves and odd numbers of nested NOT's will not.

1. h. 2. 2. 4. 3. 1. 3. 1. Delete the entry describing the NOT
2. If an entry should be deleted from the LOGIC table (example: NOT NOT A LS B)
  1. Delete the entry
  2. Set OPI = 1
  3. Make an entry in LOGIC
  4. Set OPI = 1
2. For an 'AND' or 'OR'
  1. Assign a level
  2. If there is an odd number of entries in LOGIC (under the influence of a NOT)

1. h. 2. 2. 4. 3. 2. 2. 1. Complement operator

3. Set OPI = 1

4. For statement labels, files, constants and direct code assignment operators (CATF EQ 1, 2, 3, 20)

1. Set OPI = 0

5. For subscripts (CATF EQ 7)

1. If the subscript is inactive (ACTSS EQ 0)

1. Output error #34

2. If the modifier of the subscript is ODD

1. Continue processing as a Boolean variable  
(h.2.2.4.6.1)

3. Set OPI = 0

6. For simple or subscripted items

1. If it is a Boolean item (TYPV EQ 2) or if it has an ODD modifier (CATB EQ 10)

Note: Not all Boolean expressions are assigned levels. Only those expressions which actually require testing receive levels. Boolean expressions used in simple assignment statements are not assigned levels (example: BOOL2 = BOOL1 \$). Therefore, the two procedures SRCHL and SRCHR are used to determine if the expression is used with logical operators or the IF, IFEITH, or ORIF operators. If none of these are encountered by the procedure, the Boolean expression receives no level.

1. Scan left using SRCHL for a logical operator or an IF, IFEITH, or ORIF

1. If successful

1. Assign level

2. Unsuccessful

1. h. 2. 2. 4. 6. 1. 1. 2. 1. Operate SRCHR
  1. If successful assign level
  2. Set OPI = 0
  3. If under the influence of a NOT
    1. Negate the Boolean item (CATA = 1)
  4. Delete an entry from LOGIC if required
7. For procedure labels (CATC EQ 6)
  1. If it's a Boolean procedure (TYPV EQ 2)
    1. Process as a Boolean item (h.2.2.4.6.1.)
  2. Set OPI = 0
8. For relational operators (CATC EQ 0-5)
  1. Assign a level
  2. If under the influence of a NOT (odd number of LOGIC entries)
    1. Complement the operator
    2. Set CATA = 1.

Note: Relational operators are sometimes also used as logical operators. Example:

$A \text{ LS}_{(1)} B \text{ LS}_{(2)} C$ . This expression is legal shorthand for  $A \text{ LS}_{(1)} B$  and  $B \text{ LS}_{(2)} C$ . In this example  $\text{LS}_{(2)}$  is used as both the relational operator 'Less than' and the logical operator 'AND'. If this expression were under the influence of a 'NOT', the 'AND' should be changed to an 'OR'. Therefore, CATA is set to 1 to change its meaning to 'OR'.

9. For Bead entries (CATC EQ 16).

Note: It is necessary to ignore the parenthesis

surrounding the object of a Bead. To effect this, the push-down table BEADS is used. An entry is made in this table recording the current base when a Bead (CATC EQ 16) is encountered. If a parenthesis is encountered and the level at that time matches the level last recorded in the BEADS table, that parenthesis is concluded to surround the object of a Bead and is therefore ignored. Similarly the right parenthesis is identified and causes entries to be deleted from the BEADS table.

1. h. 2. 2. 4. 9. 1. Make an entry in the BEADS table

2. Set OPI = 0

10. Bracket processing according to the form (CATF) of the entry by activating the switch BRK

1. For left parenthesis (CATF EQ 0, 10)

1. If this is the left parenthesis associated with the object of a Beaded expression (BASE EQ BDBSE)

1. Set CATF = 10

2. Continue at h.2.2.4.10.1.3.

2. Increase the base by a unit (20)

3. Make a zero entry into the comma table (CIND = 0)

4. Set OPI = 1

2. For right parenthesis (CAT EQ 0, 11)

1. If this is the right parenthesis associated with the object of a Beaded expression (BASE EQ BDBSE)

1. Set CATF = 11

2. Delete an entry from the BEADS table

3. Continue at h.2.2.4.10.2.3.



1. h. 2. 2. 4. 10. 2. 2. Decrease the base by a unit (20)
3. Delete an entry from the COMMA table
4. Delete an entry from the LOGIC table if required.
5. Set OPI = 0
3. Left bracket (CATF EQ 2, 8).

Note: The general form of a subscript expression which can be contained in a single IL subscript entry is variable  $\pm$  constant. This expression and variations of it (as listed below) are called the standard subscript forms. Since all of the elements of one of these forms can be contained in a single IL entry, it is not necessary to assign levels within one. Also, if subscript expressions are nested and the nested expression is of standard form, the nested expression is entered into the IL using a special nester operator (ILO = 26). Therefore, when a left bracket is encountered, the procedure SSPRO is used to analyze the subscript expression. The purpose of the analysis done by this procedure is to determine when it is necessary to assign levels to arithmetic operators used within subscript expressions and when it is necessary to assign nester levels to subscript variables or beaded expressions located within brackets. Expressions located within subscript brackets or within commas within subscript brackets are deemed to be of standard form when they are of the following type or when they can be reduced to the following types:

$\pm$  VAR or SUBS or TEMP

$\pm$  VAR or SUBS or TEMP  $\pm$  CONS

$\pm$  CONS

$\pm$  ABS (VAR or SUBS or TEMP)

ABS (VAR or SUBS or TEMP)  $\pm$  CONS

with VAR meaning variable

SUBS meaning subscript

TEMP meaning temporary storage

CONS meaning constant

**Examples:**

Let SV mean subscripted variable

SV[A+1]: A+1 standard

SV[A+1\*B]: A+1\*B not standard

SV<sub>1</sub>[SV<sub>2</sub>[A]+1]: SV<sub>2</sub>[A] standard-assign nesting  
level reducing the expression  
to:

SV[temp+1]: Temp+1 standard

BYTE[A+1,B+1](VAR): A+1 standard

B+1 standard

SV[BYTE[A,B](VAR)]: BYTE[A,B](VAR): standard-  
assign nesting level reducing  
the expression to:

SV[TEMP]: standard

The SS table is used in a 'push-down' manner in the analysis, thus permitting nested expressions to be considered for standard form. Example:

SV<sub>(1)</sub>[SV<sub>(2)</sub>[SV<sub>(3)</sub>[VAR]+1]+2]

The analysis of the expression starting with bracket (1) is started but can not be completed until the expression starting with (2) is complete. This, in turn, can not be completed until (3) is complete. All of the expressions are of standard form. Items in the SS table are used to remember which parts of the standard forms have been recognized in a given expression. If the right boundary (either a comma or right bracket) of a potential standard subscript expression is encountered before a duplicate or illegal standard form part of speech is discovered, the expression is considered to be of standard form.

1. h. 2. 2. 4. 10. 3. 1. Increase the base by 1 unit (20)
2. Prepare to use the next entry in the SS table
3. Make a bracket entry in the COMMA table (CIND = 1)
4. Analyze the subscript expression to see if it is of standard form using procedure SSFRO

1. h. 2. 2. 4. 10. 3. 4. 1. If not of standard form (STDI EQ 0)
  1. Set OPI = 1
  2. Continue scan at h.2.2.4.
2. If standard form (STDI EQ 1) and the expression terminated with a comma (COMI EQ 1)
  1. Continue at h.2.2.4.10.4.3.1.
3. If standard form (STDI)
  1. Process the right bracket ending the expression at h.2.2.4.10.4.
4. Right bracket (CATF EQ 3)
  1. Assign a level of comma (priority of 1) (This level is used as an Anchor point level only so that an entire subscript expression will be processed prior to advancing further thru the statement. When the entire subscript expression is processed, this level is removed.)
  2. Decrease the base by 1 unit (20)
  3. If there is only one entry in the SS table (there are no nested subscript expressions) or the type of the preceding expression was arithmetic (STYPE EQ 0)
    1. If the expression being processed is terminated by a comma (COMI EQ 1)
      1. Assign a level to the comma
      2. Clear the SS entry except for the CAT entry number of its left bracket (LBPOS)
      3. Continue processing as if the comma were a left bracket (h.2.2.4.10.3.4.)
    2. Delete an entry from the SS table and the COMMA table
    3. Continue processing at h.2.2.4.

1. h. 2. 2. 4. 10. 4. 4. If the preceding expression was a Bead type (STYPE EQ 1)
  1. Set the form (CATF) of the left bracket commencing the bead expression to 8
  2. Assign a nest level (12) to the left bracket commencing the bead expression (nested bead expression of standard form)
  3. Continue processing at h.2.2.4.10.3.2.
5. Using the procedure SSFRO, determine if the preceding expression is of standard form (example:  $SV[SV[A_{(1)}] + 1]$  ) processing is now at the right bracket (1)
  1. If the expression is standard (STDI EQ 1)
    1. If the preceding expression contained an ABS modifier
      1. Reduce the base by 1 unit (20)
    2. If the expression terminated in a comma (COMI EQ 1)
      1. Delete an entry from table SS
    3. If the expression terminated in a right bracket
      1. Set POS to POS-1 so the right bracket will be processed
    4. Assign a Nest level (12) to the left bracket of the expression
    5. Continue processing at h.2.2.4.10.3.2.
  2. The expression is not of standard form (STDI EQ 0)
    1. Reset the type of the preceding expression to arithmetic (STYPE = 0)

1. h. 2. 2. 4. 10. 4. 5. 2. 2. Assign a comma level (1) to the left bracket of the preceding expression (this is only an anchoring level and after it is once used it is eliminated)
3. Change the form (CATF) of the left bracket to 12
4. Delete an entry from the SS and COMA tables

### 3. Level Analysis

1. General Philosophy - This phase makes a scan thru the statement converting the statement into intermediate language in an order determined by an analysis of the levels assigned by the first phase. A right scan is made thru the statement comparing the level of each operator with the level of the operator immediately preceding it (to the left of it). If an operator is encountered with a level which is less than or equal to the level of the operator to its left, the operator on the left is entered into the IL table, a left scan is begun to determine if the 'anchor-point' can still be considered an 'anchor-point' with respect to the next operator to its left. As long as the 'anchor-point' relationship exists, operators and their operands are put into the IL. If a level is encountered which is less than the level of the 'anchor-point', the right scan is initiated once more searching for a new 'anchor-point'. The end of a statement is considered as the final 'anchor-point' and when this point is determined, each operator encountered in the left scan is immediately entered into the IL.

Two other general operations are performed in this phase. The first is the assignment of intermediate or temporary working storage. Each time an operator and its operands are entered into the IL an intermediate result is determined. If this intermediate result is one of the operands for the next operator entered into the IL, no specific assignment to a temporary storage is necessary. However, if it is not used immediately, an assignment to a particular temporary storage is required. The general rule for determining an assignment is: when a new 'anchor-point' is required, the last 'result' implied in the IL must be assigned a temporary storage.

The other general operation is the determination of transfer points for relational and logical expressions. It is necessary to determine the linkage between the various parts of a relational statement. This is accomplished using the procedure XFRPT and is explained in more detail in that region of the procedure directly concerned with relational operators.

1. h. 3. 2. Specific Operations

1. If the value of the base is not 0 (BASE NQ 0) meaning there was an odd number of parentheses or brackets in the statement
  1. Output error #28
  2. Exit
2. Housekeep the items required in this phase.
3. Initiate the scan at the CAT table entry specified in FIRST.
4. Locate a CAT table entry containing a level (CATE NQ 0) or else the end of the statement (the entry specified in LAST).
5. Determine if the entry with the level is an 'anchor-point' using procedure LVLCP.
6. If the last assignment to temporary storage was unnecessary (BNWS EQ 1)
  1. Change the last assignment to specify a temporary storage of 0 or result.
  2. Reduce the count of temporary storage assignments by 1 (NWS = NWS-1).
7. If an 'anchor-point' could not be determined by LVLCP because the beginning of the statement was encountered prior to the locating of an operator with a level (LEVEL EQ 3)
  1. Continue the right scan at h.3.2.4.
8. If the conditions for an 'anchor-point' were not met (LEVEL EQ 0)
  1. If an assignment to a temporary storage should be made (RSLT NQ 0)

1. h. 3. 2. 8. 1. 1. Initialize item RSLTI to 1 to indicate "the number of operators + 1" that have been processed since the last temporary storage assignment.
2. Generate a new temporary storage "register" (NWS = NWS+1)
3. Remember the entry in which the assignment is being made in case it was not really necessary (POLOC = OLOC) (See h.3.2.6).
4. Make the assignment in the last IL operator entry (OLOC)
5. If the last operator required a temporary storage operand entry as its first (left) operand (ASWTP EQ 1) (for example, an assignment of a nested Bead expression)
  1. Update the assignment in the operand entry.
6. Substitute the new temporary storage for the result of the last operator in the CAT table.

Note: As operators and operands are entered into the IL they are cleared from the CAT table and for each operator, an entry describing a temporary result (CATC = 18) is inserted. This temporary result entry represents the result of performing the operation specified by the operator on its operands. Therefore, this entry must also be changed when a temporary storage assignment is made for this operator. Item RSLT contains the CAT table entry number of the last temporary result entry substituted for an operator.

7. If this is a forced assignment of temporary storage (RIGHT NQ R)

Note: There are certain cases which require an assignment of a temporary storage even though the result might be used immediately. The first case is arithmetic expressions used in parameters for procedure calls. No IL operand entry describing a parameter or subscript or bead entry associated with a parameter can specify a temporary storage of result (ILD EQ 3, ILF EQ 0). (This is a

Translator restriction primarily applicable to output parameter expressions.) The second case is in relational statements containing implied AND's. Example:

A LS B+C LS D

In this example B+C must be assigned to temporary storage because it is used as an operand for more than one operator. These two cases are recognized by procedure LVLCP. It adjusts the scan to force the temporary storage assignment by setting RIGHT = RIGHT-1 causing the current operator to be examined twice; the first examination forces the assignment, the second examination proceeds normally.

1. h. 3. 2. 8. 1. 7. 1. Step RSLTI by 1 to prevent the later deassignment of the temporary storage (see h.3.2.6.).
2. Continue the right scan (h.3.2.4.)
9. If end-of-statement was determined (LEVEL EQ 2)
  1. If no operators were determined in the left scan (LEFT LS FIRST) and the end-of-statement is not an 'anchor-point'
    1. Perform a right scan starting at the beginning of the statement (RIGHT) looking for an entry describing the last possible operand in the statement.
      1. If the scan is successful, use the procedure PUTIN to enter the operand in the IL table leaving an empty entry for an operator entry.
    2. Exit
  2. If an operator was encountered (meaning the end-of-statement is an 'anchor-point') continue processing at h.3.2.10.1.
10. If an 'anchor-point' was determined (LEVEL EQ 1)
  1. Step RSLTI by 1 to indicate the number of operators +1 processed since the last temporary storage assignment.



1. h. 3. 2. 10. 2. If the class of the operator on the left is separator (CATC EQ 9) implying an = or == operator
  1. Generate an IL operator entry containing the appropriate operator (ILO = 21,22)
  2. Process the operands as arithmetic operands at h.3.2.10.4.4.
3. If the class of the operator on the left is bracketer (CATC EQ 10) implying a nesting operation
  1. If the bracketer is of form 8(CATF EQ 8) meaning this is the left bracket of a bead
    1. Generate an assignment IL operator entry (ILO = 21)
    2. Generate an operand entry of temporary storage specifying result (ILF = 0)
    3. Using the procedure BDSV enter the second operand.
  2. If the bracketer is not of form 8(CATC NQ 8) it must be the left bracket of a subscript expression.
    1. Generate a nester IL operator entry (ILO = 26)
    2. Locate the operand associated with the bracket using procedure POSOD
    3. Enter the operand into the IL using procedure PUTIN
  3. Exit to continue the scan (h.3.2.4)
4. If the class of the operator on the left is arithmetic (CATC EQ 13)
  1. Generate an arithmetic IL operator entry (ILO = 7-12)
  2. If the form of the entry is 'OF' (CATF EQ 5) meaning a procedure call

1. h. 3. 2. 10. 4. 2. 1. Force any previous assignments to temporary storage to be maintained by setting RSLTI = 3.
2. If a procedure name is not described in the entry preceding the OF
  1. Output error #32
  2. Exit at the error exit
3. Generate an IL operand entry containing the name of the procedure.
4. Set CALL to the dictionary entry number of the procedure.
5. Record the current level of working storage at the procedure call using procedure WORST.
6. Enter the parameters in the IL.
  1. If no parameters (the OF is followed by a \$ or empty brackets ( ) ), continue at 7.
  2. Keep a count of the parameters in NOPAR.
  3. Set IOPI to 0 for input parameters and to 1 when the '=' is encountered indicating output parameters follow.
  4. Set ILR to IOPI in each IL entry (specified in ODPOS) describing a parameter.
  5. If the parameter is an express item (DEFN EQ 1) and it is an output parameter (IOPI EQ 1) and this call does not occur in the main body of a program (PRCD NQ 9000), set PDAT = 0 indicating an express item is being set.
7. Enter the count of the parameters (NOPAR) in ILE of the first IL operand entry of the call.
8. If the procedure is a function (SIND EQ 1)
  1. Continue the scan at h.3.2.4.

1. h. 3. 2. 10. 4. 2. 9. Operate the procedure LLOC to determine if there are any other operators present in the statement.
  10. If there are other operators in the statement, or the 'anchor-point' is not on the \$ (meaning the procedure is used as a function)
    1. Output error #32
    2. Exit to the error exit
  11. Exit
3. If the form of the entry is ABS or unary - (CATF EQ 7,6)
  1. Perform a right scan to locate the operand associated with the operator.
  2. If the operator is unary - :complement CATA
  3. If the operator is ABS: set CATB to 6 and CATA to 0.
    1. If the operand is the object of a Bead expression
      1. Output error #30
    4. Exit to continue the right scan at h.3.2.4.
  4. Locate the position of the left operand using procedure POSOD.
  5. Enter the left operand using procedure PUTIN.
  6. Enter the right operand using procedure PUTIN.
  7. Exit to continue the scan at h.3.2.4.
5. If the class of the operator on the left is file, simple item, subscripted item, procedure, subscript or temporary storage (meaning the operand is Boolean or is modified by a Boolean modifier) (CATF EQ 2,4,5,6,7,18)

Note: The normal structure of a Boolean test in the intermediate language is 'BOOL NQ 0'. Therefore, when processing a Boolean expression the operator 'NQ' must be supplied as well as the operand '0'.

1. h. 3. 2. 10. 5. 1. Generate a 'not equal' IL operator entry (ILO = 5).
2. If the operand is under the influence of a NOT (CATA EQ 1)
  1. Complement the IL operator (ILO = 2).
3. Enter the Boolean operand into the IL using procedure PUTIN.
4. Generate an IL operand entry specifying the operand 0 (DICT entry 1 specifies the constant 0).
5. Continue processing as a relational statement at h.3.2.10.6.6.

6. If the class of the operator on the left is relational operator (CATC EQ 12)
  1. Generate a relational IL operator entry (ILO = 0-6).
  2. Locate the left operand using procedure POSOD.
  3. Enter the left operand into the IL using procedure PUTIN.
  4. If a relational operator is described in the 'anchor-point' entry (CATC EQ 12)

Note: An expression of the form A LS B LS C is being processed when an 'anchor-point' describes a relational operator and the operator to its left is also a relational operator. Since this is really short hand for the expression A LS B AND B LS C, it is necessary that the middle operand, B in this example, not be erased when it is first entered into the IL. (Customarily, CAT table entries are cleared as they are converted into IL.) Also since the expression may have been originally of the form A LS B+D LS C, it is necessary that the expression B+D be assigned permanently to temporary storage.

1. Set ERAS = 1 to stop the right operand from being erased.
2. Set RSLTI = 3 to prevent the elimination of a temporary storage assignment.

1. h. 3. 2. 10. 6. 5. Enter the right operand into the IL using procedure PUTIN.
6. Using procedure LLOC locate the first entry containing a level which precedes the 'anchor-point' entry.
7. Determine the transfer point required for the expression using procedure XFRPT.

Note: Of primary concern in the analysis of logical expressions is the determination of the linkage required between the various portions of the expressions.

Example:

If  $A \wedge B \wedge C \vee$   
     $D \wedge E \wedge F \vee$   
     $H \wedge I \$$

In the example, the transfer points for A and B should be false transfer points and should point to where D is tested (internal false transfer points). Similarly, D and E should point to where H is tested. The transfer points for C and F should be true transfer points pointing to the end of the statement. The transfer points for H and I should be false transfer points pointing to the false transfer point of the entire statement (external false transfer point).

Therefore, as part of the analysis, the linking transfer points must be determined and assigned labels and the operators must be adjusted (complemented) if necessary to effect the meaning of the statement. The function of procedure XFRPT is to carry out this part of the analysis. In the discussion of the rules used by XFRPT in its analysis let:

TTP mean the true transfer point for the entire expression.

FTP mean the false transfer point for the entire expression.

LTP mean a link transfer point internal to the expression.

The algorithm for the determination of the linkage necessary for relational statements is:

- (1) If the 'anchor-point' indicates end-of-statement:
  - (a) Complement the relational operator first entered in the IL.
  - (b) Use FTP as its transfer point.
- (2) Initiate a right forward scan from the 'anchor-point' searching for a logical operator (AND or OR) whose level is less than the level of the 'anchor-point'.
  - (a) If the end-of-statement is encountered
    - (1) If the 'anchor-point' is AND, complement the last relational operator and use FTP as the transfer point.
    - (2) If the 'anchor-point' is OR, use TTP as the transfer point.
  - (b) If the scan successfully locates an operator which is identical with the 'anchor-point':
    - (1) The level of this new operator is substituted for the level of the 'anchor-point' and the scan continues.
  - (c) If the scan successfully locates an operator which is different from the 'anchor-point':
    - (1) If the 'anchor-point' is an AND, the relational operator last entered into IL is complemented.
    - (2) LTP is used as the transfer point for the last relational entered in the IL.
    - (3) The LTP must be entered into the CAT table following the operator located by the scan so that it will be entered into the IL at the proper location.

(The TTP and FTP are entered into the IL in the normal course of processing; see main-program, region L080.) (Incidentally, when the CAT table is made up, an empty entry is included following each entry describing an AND or OR. It is in this entry that the LTP is saved.)

1. h. 3. 2. 10. 6. 8. If the 'anchor-point' is end-of-statement
  1. Continue processing at h.3.2.10.6.12.
9. If the 'anchor-point' is a logical operator (and not a relational operator substituting as a logical operator) (ERAS EQ 0)
  1. Erase the entry containing the AND or OR.
10. If there was a label saved by procedure XFRPT in the entry succeeding the logical operator
  1. Enter the label into the IL using procedure STLE.
11. Continue the right scan at h.3.2.4.
12. If the scan performed by LLOC (at 6.) was unable to locate other operators in the statement (LEFT LS FIRST)
  1. Exit
13. If LLOC did not locate a CAT table entry describing an '='
  1. Output error #31
  2. Exit to the error exit.
14. If LLOC did locate an entry containing the operator '=' (meaning the expression being processed is the last portion of a Boolean assignment statement)

Note: The structure of a Boolean assignment statement in the intermediate language is as if the following JOVIAL statements had been processed at the end of the relational portion of the assignment statement:

```
TTP.  BOOL = 1 $  
      GOTO EXIT'LABEL $  
FTP.  BOOL = 0 $  
EXIT'LABEL.
```

1. h. 3. 2. 10. 6. 14. 1. If the statement has a TTP label (ATTP NQ 0)
1. Enter the label using STLE.
  2. Generate an assignment IL operator entry (ILO = 21).
  3. Locate the position of the Boolean operand using procedure POSOD.
  4. Enter the operand using procedure PUTIN and generate an operand entry describing the constant 1.
  5. Generate a label to serve as the EXIT'LABEL.
  6. Generate a GOTO IL operator entry.
  7. Generate an IL operand entry describing the EXIT'LABEL.
  8. Set BV = OLOC for possible later optimizing of the GOTO entry.
  9. Enter the FTP of the statement using procedure STLE.
  10. Generate an assignment IL operator entry (ILO = 21).
  11. Enter the Boolean operand using procedure PUTIN and generate an operand entry describing the constant 0.
  12. Enter the EXIT'LABEL using procedure STLE.
  13. Exit.



10 July 1962

227

TM-555/020/00

1. h. 3. 2. 11. Exiting procedure

1. If only one operator was processed since the last temporary storage assignment (BNWS EQ 1 and RSLTI LS 3)
  1. Alter the last temporary storage assignment to specify result (ILF = 0).
2. If the amount of temporary storage exceeds that of previous statements (MAXWS LS NWS)
  1. Set MAXWS = NWS
3. Exit.

2. a. Name - BDSV
- b. Description - This procedure generates IL operand, bead and subscript entries from CAT table entries describing a beaded or subscripted variable expression.
- c. Parameters
  1. Input
    1. LLCAT - specifies the starting CAT table entry number of the expression to be processed.
    2. BDI - indicates whether a beaded or subscripted variable expression is to be processed.  
  
1 = byte expression  
2 = bit expression  
0 = subscripted expression
    3. DELE - indicates whether the CAT table entries examined are to be cleared.  
  
1 = do not clear entry  
0 = clear entry
  2. Output
    1. EEEE - error exit
- d. Local items
  1. COMAS    4. BD        7. FIL
  2. NOBDS    5. SUBV    8. LIL
  3. LCAT     6. PREN    9. ABSI
- e. Express Items
  1. NIL        4. BNWS
  2. NWS        5. CAT table items
  3. RSLTI      6. IL table items
- f. Procedures called
  1. ERROR

## 2. g. Used by

1. ANCHR
2. PUTIN

## h. Operations

Note: The procedure recognizes from context when it is processing the subscripted or beaded variable and when it is processing the subscript or bead expression associated with the variable. Examples:

SV[A+1] - SV is the subscripted variable

- A+1 is the subscript expression

BYTE[A,B](VAR) - VAR is the beaded variable

- A,B is the bead expression

BIT[A+1,B](SV[C]) - SV is both a subscripted and a beaded variable

- A+1,B is the bead expression

- C is the subscript expression

Furthermore, the procedure expects all subscript and bead expressions to be or to have been reduced to standard subscript form (see discussion of standard subscript form in procedure ANCHR). To simplify the processing of the various combinations included in the standard subscript, the procedure sets up a typical IL subscript-bead entry having the structure of 0 + 0. This basic structure or shell is filled in and altered according to the actual expression processed.

1. If processing a beaded expression (BD NQ 0)

1. Clear the CAT table entry containing bead.

2. Examine each entry in the expression by activating the switch ODCLS.

1. For entries containing bracketers, analyze using switch BRK.

1. Ignore simple parenthesis (CATF EQ 0,1).

2. For simple left brackets (CATF EQ 2) or special left brackets (CATF EQ 8,12).

1. If processing a subscript expression (BD EQ 0)

1. Set up a typical IL subscript entry (ILF=ILE=1, ILC=2).

2. h. 2. 1. 2. 2. If processing bead expression (BD EQ 0)
  1. Set up a typical IL bead entry (ILF=ILE=1, ILC=3)
3. For right brackets (CATF EQ 3)
  1. If processing a bead expression (BD NQ 0)
    1. If the 'number of characters' portion of the expression is absent from the expression implying a 'number of characters' of 1 (COMAS EQ 0)
      1. Generate a bead entry specifying a 'number of characters' of 1 by
        1. Set NOBDS = 1 (DICT entry of constant 1).
        2. Set LCAT = LCAT - 1 (so that the right bracket entry is processed twice).
        3. Continue processing at h.2.7.2. (as if the right bracket were a comma).
      2. Set NOBDS = 1 (DICT entry of constant 0).
    2. If processing a subscript expression (BD EQ 0)
      1. If not processing the beaded variable (PREN EQ 0)
        1. Exit
  4. For special left parenthesis (CATF EQ 8) (the left parenthesis bracketing a beaded variable)
    1. Set BD = 0 so the procedure will cease generating bead entries.
    2. Set SUBV = 1 indicating the procedure is now processing the beaded variable.
    3. Set PREN = 1 to acknowledge recognition of the left parenthesis.
    4. If more than 2 bead factors have been encountered (COMAS GR 1)
      1. Output error #36

2. h. 2. 1. 4. 4. 2. Eliminate extra bead factors from IL table.
5. For special right parenthesis (CATF EQ 9) (the right parenthesis bracketing a beaded variable)
  1. Set PREN = 0 acknowledging recognition of the right parenthesis of a beaded variable.
  2. Exit
2. For entries describing statement label (CATF EQ 1) or subscripted variables (CATF EQ 5)
  1. Generate an IL operand entry from the information in the CAT table entry.
3. For entries describing simple variables (CATF EQ 4)
  1. If processing the beaded variable (SUBV EQ 1)
    1. Generate an IL operand entry.
    2. If an ABS modifier has been encountered (ABSI EQ 1)
      1. Do not set ILB from CATB (already set to ABS).
  2. If processing a bead or subscript expression
    1. Generate a bead or subscript IL entry.
4. For entries describing temporary storage (CATC EQ 16)
  1. Change the temporary storage to result if possible (CATF EQ NWS and RSLTI LS 3).
  2. Process as simple variable at h.2.3.
5. For entries describing constants (CATC EQ 3)
  1. Set ILE of the bead-subscript typical entry to CATF of the constant.
6. For entries describing a subscript (CATC EQ 7)
  1. If processing a subscript expression (BD EQ 0)
    1. Set SUBUI = 1 (indicating the subscript was used as a subscript)

2. h. 2. 6. 2. Complete the typical bead-subscript IL entry.
7. For entries describing separators (CATC EQ 9)
  1. If separators other than ',' or '...' (CATF EQ 1,5)
    1. Output error #32
    2. Exit to the error exit
  2. For a ',' or '...'
    1. Initialize a typical bead or subscript entry (ILF = ILE = 1).
    2. Maintain a count of the number of such separators in item COMAS.
8. For entries describing arithmetic operators (CATC EQ 13)
  1. For a minus or unary minus (CATF EQ 1,6)
    1. Set ILA in the typical bead-subscript entry to 1.
  2. For an ABS (CATF EQ 7)
    1. Set ILB in the typical bead-subscript entry to 6.
    2. Set ABSI = 1 to acknowledge recognition of an ABS.
9. For all other classes of entries
  1. Output error #32
  2. Exit to the error exit

10 July 1962

233

TM-555/020/00

3. a. Name - BTOD

b. Description - This procedure converts a binary number into standard transmission code.

c. Parameters

1. Input

1. AA - contains the value to be converted.

2. Output

1. BTOD - contains the converted number.

g. Used by

1. PRTIL	4. PRISR	7. MP
2. PPTCT	5. ERROR	
3. PRTQ	6. WORST	

h. Operations

1. Using direct code converts binary numbers to STC.

4. a. Name - DESIG
- b. Description - This procedure analyzes and converts into IL switch point expressions or GOTO expressions.
- c. Parameters
  1. Input
    1. PI1 - Specifies the type expression to be processed.  
  
0 = GOTO statement  
1 = numerical switch  
2 = item switch
    2. PI2 - Specifies the beginning entry number in the CAT table of the expression to be processed.
  2. Output
    1. PO1 - Specifies the terminal entry in the CAT table of the expression processed.
    2. PO2 - Error exit
- d. Local items
  1. T1
- e. Express items
  1. NIL      4. CAT Table Items
  2. FIRST    5. IL Table Items
  3. LAST     6. DICT Table Items
- f. Procedures Called
  1. LNETH
  2. STLE
  3. ANCHR
  4. OPND
- g. Used by
  1. MB



## 4. h. Operations

1. If the first entry examined <sup>1-5</sup> is a comma or a right parenthesis (meaning blank switch point)
  1. If PIL is not zero, in other words we are processing a switch point
    1. Enter a generated label of zero in the IL.
  2. Exit
2. If a switch point is being processed (PIL not equal to zero)
  1. Activate the switch SW to process the point by its dictionary class (CLAS).
    1. For statement labels, close labels or compool program labels (CLAS EQ 1,6,11)
      1. Enter the label into the IL using the procedure OPND.
      2. Set OLAY in the dictionary if the point being processed is a statement label. (This indicates the label was used in a switch.)
      3. Clear the class CATC of the entry.
    2. For switch labels specified as switch points (CLAS EQ 9,10)
      1. Enter a generated label in the IL using the procedure STLE
      2. Save the label in the CAT table.(CATE).
      3. Determine the right boundary of the expression using the procedure LENGTH.
        1. Save the number of entries of the expression in the CAT table. (CATF).
        2. Set P4 from this value.
    3. For Procedure labels (CLAS EQ 6)
      1. Output an error message.
      2. Exit.

4. h. 2. 1. 4. For all other classes
  1. Output an error message
  2. Exit
2. If the switch being processed is a numeric switch (PIL EQ 2)
  1. Put the constant associated with the point into ILE (the channel of the constant is specified in the item PI2).
  3. Exit.
3. If processing a GOTO (PIL EQ 0)
  1. If the class of the entry (CATC) is zero
    1. Exit (means this entry has already been processed as a switch point).
  2. If a label is associated with the entry (See h.2.1.2.2.)
    1. Enter the label into the IL using procedure STLE.
  3. Make up an IL GOTO entry.
  4. Activate the switch SW1 to process the CAT table entry by its dictionary class (CLAS).
    1. For statement labels, close labels and compool program labels (CLAS EQ 1,11,12) enter the appropriate label into the IL using the procedure OPND.
    2. For switch labels (CLAS EQ 9,10)
      1. Simple switch expressions are inserted into the IL using procedure OPND.
      2. Complex expressions are processed into the IL using procedure ANCHR (the length of the expression is obtained from the CAT table, see h.2.1.2.3.1.)
    3. For procedure labels (CLAS EQ 6)
      1. Output an error message
      2. Exit

10 July 1962

237

TM-555/020/00

4. h. 3. 4. 4. For all other classes of entries

1. Output an error message
2. Exit

## 5. a. Name - ERROR

b. Description - This procedure composes an error message into item IMAGE.

## c. Parameters

## 1. Input

1. PIL - contains the number of the error message.

## e. Express items

1. IMAGE
2. PROCN
3. LPLUS
4. LNAME
5. ERCNT

## f. Procedures called

1. BTOD
2. NTOI
3. PRNT

## g. Used by

- |           |           |           |
|-----------|-----------|-----------|
| 1. MP     | 5. WORST  | 9. PUTIN  |
| 2. FACT   | 6. ANCHR  | 10. EDSV  |
| 3. DESIG  | 7. SSFPRO | 11. XFRPT |
| 4. LENGTH | 8. POSOD  |           |

## h. Operations

1. Put standard error message format into IMAGE.
2. Put the error number plus 100 into IMAGE. (The addition of 100 differentiates Gen2 errors from Gen1.)
3. If error occurred during the processing of a procedure (PROCN NQ 0)
  1. Adjust IMAGE to reference the procedure specified in PROCN.
4. Put the name of the last program label encountered (LNAME) and the number of statements processed since that label (LPLUS) into IMAGE.
5. If 200 errors or more
  1. Put error #139 in IMAGE.
  2. Output IMAGE using PRNT.
  3. Exit to terminate compilation.
6. Output image using PRNT.

## 6. a. Name - FACT

b. Description - This procedure analyzes the expressions or factors of a FOR statement and converts the expression into IL.

## c. Parameters

## 1. Input

1. P1 - Contains the CAT table entry number of the starting position of the section of IL to be processed.
2. P2 - Contains the value of the operator (LOD, INC, TST) which is to be inserted in the IL.

## 2. Output

1. P3 - Indicates the delimiter at the end of the expression

1 = comma  
0 = \$

2. P4 - Specifies the terminating entry in the CAT table of the expression precessed.

3. P5 - Error exit

## d. Local items

1. T3

## e. Express items

- |         |          |       |
|---------|----------|-------|
| 1. NIL  | 4. FIRST | 7. IL |
| 2. OLOC | 5. LAST  |       |
| 3. CAT  | 6. SRTD  |       |

## f. Procedures called

1. OPND
2. ANCHR

## g. Used by

MB

## 6. h. Operations

1. The boundaries of the expression are determined.
  1. A forward scan is performed.
    1. The pushed down counter T3 is used to recognize when the scan is inside or outside bracketed expressions.
    2. Entries within bracketed expressions are ignored.
    3. The scan is terminated on a comma or \$ (encountered outside of a bracketed expression).
    4. Output item P3 is set to a one if the terminating symbol is a comma, or a zero if the terminating symbol is a dollar sign.
    5. If 250 entries are examined without success
      1. Output error
      2. Exit
  2. The isolated expression is converted into IL.
    1. Procedure OPND is used to put simple expressions into the IL.
    2. Procedure ANCHR converts complex expressions into IL.
  3. The operator supplied in P11 is entered into the IL entry specified by the item OLOC. (This item is set by the procedure itself if it puts the operand in using OPND; otherwise the item is set by ANCHR.)

10 July 1962

241

TM-555/020/00

7. a. Name - HOLLS
- b. Description - This procedure converts standard transmission characters into Hollerith.
- c. Parameters
  1. Input
    1. TC - contains the characters to be converted.
    2. NO - contains the number of characters to be converted.
  2. Output
    1. HL - contains those characters of TC which were converted.
- d. Local Item
  1. TPOH
- g. Used by PRNT
- h. Operations
  1. Starting with character 0 of TC, convert the number of characters of TC specified by NO into Hollerith code locating the converted characters in HL starting at character position 0.

10 July 1962

241

TM-555/020/00

7. a. Name - HOLLS

b. Description - This procedure converts standard transmission characters into Hollerith.

c. Parameters

1. Input

1. TC - contains the characters to be converted.

2. NO - contains the number of characters to be converted.

2. Output

1. HL - contains those characters of TC which were converted.

d. Local Item

1. TTOH

g. Used by PRNT

h. Operations

1. Starting with character 0 of TC, convert the number of characters of TC specified by NO into Hollerith code locating the converted characters in HL starting at character position 0.



## 8. a. Name - IFO

- b. Description - This procedure generates IL 'Information' type entries which supply the boundary IL entry numbers of a relational statement referring to the label preceding the 'Information' entries.

## c. Express Items

1. IL table items
2. DRQ table items

## g. Used by

1. MP

## h. Operations

1. Generate the 'Information' operator and operand IL entries locating the entries at the entry specified by NIL+1 (this facilitates the automatic erasing (by new entries added to the IL table) of the 'information' entries when they are not required).

## 9. a. Name - ILDF

b. Description - This procedure converts the code (ILD) and channel (ILF) of an IL entry into a format for outputting.

## c. Parameters

1. ILFF - contains the IL entry number of the entry to be processed.
2. FIRST - contains the starting character position in IMAGE into which the formatting is to be done.

## e. Express items

1. IMAGE
2. IL table items
3. STSD
4. STID

## f. Procedures called

1. NTOI
2. BTOD

## g. Used by

1. PRTIL

## h. Operations -

1. If the code is dictionary (ILD EQ 0)
  1. Format the name using procedure NTOI.
2. For all other codes
  1. Put the actual code name into IMAGE
  2. If the code is subscript (ILD = 2)
    1. Put the name of the subscript into IMAGE
  3. For all other codes
    1. Put the value of ILF directly into IMAGE

## 10. a. Name - INBQ

b. Description - This procedure makes entries in that part of the DRQ table reserved as the temporary push-down queue (entries 1 and 2). The entries are made depending on the type of delayed-processing required by certain statements.

## c. Parameters -

## 1. Input

1. PI1 - specifies the type of statement requiring delayed-processing.

0 = FOR statement	3 = ORIF statement
1 = Procedure or Close declaration	4 = IFEITH statement
2 = IF statement	

2. PI2 - the content depends on the type of entry (PI1)

PI1 EQ 0 -specifies the SRT channel number associated with the FOR

PI2 EQ 1 -specifies the exit label for the procedure or close

PI2 EQ 2,3,4 -specifies the false transfer point associated with the statement

## e. Express items

1. NEST	4. NIL	7. STMC
2. POLOC	5. NDRQ	8. DRQ table items
3. LOC	6. STPC	

## g. Used by:

1. MP

## h. Operations

Note: In order for the DRQ push-down list to function properly, it is necessary that entries be made in a temporary push-down queue before being put into the normal push-down list. This permits statements requiring delayed-processing to be used successively and yet to have their delay-processing performed one statement later. (If it were not done this way, the second statement requiring the delayed-processing would cause the first statement to be pushed down in the list and not to come out until the second statement's delayed-processing was completed.)

The use of the temporary queue also allows a statement to be used immediately preceding the END terminating a compound statement.

10. h. 1. Clear next DRQ table entry
2. If the type of entry is FOR (PIL EQ 0)
  1. Set DRQE = 1 to initialize the count of the number of FOR statements associated with this DRQ entry.
  2. Step NEST to keep a running count of the number of active subscripts.
3. If the type of entry is to be procedure or close (PIL EQ 1)
  1. Set DRQI = GLAB, the exit label for the procedure or close.
  2. Set PLOC to the DRQ entry number of the entry currently being filled.
4. If the type of entry is to be IF, IFEITH, ORIF (PIL EQ 2,3,4).
  1. Set DRQI = LOC. (LOC contains the IL entry number of the first IL entry associated with the IF, IFEITH, ORIF statement.)
5. Complete the DRQ entry with the standard items.
6. If the entry just made in the DRQ table has a type of IF (PIL EQ 2), the preceding entry has a type of IF (DRQA EQ 2), there have been no intervening statements processed between the making of the two DRQ entries (DRQC EQ DRQC - 1) and the two statements associated with the DRQ entries are in the same compound statement (DRQB EQ STPC) meaning there were two IF statements in a row)
  1. Exchange the two DRQ entries so that the delayed-processing will be performed at the right time.

- 11. a. Name - LLOC
- b. Description - This procedure makes a backward scan thru the CAT table searching for an entry having a level (CATE not equal to 0).
- c. Parameters
  - 1. Input
    - 1. RLOC - Beginning entry number in the CAT table for the scan.
  - 2. Output
    - 1. LLO - Entry number of the entry at which the scan was terminated.
- e. Express items
  - 1. FIRST
  - 2. CATE
- g. Used by:
  - 1. ANCHR
  - 2. LVLCP
- h. Operations
  - 1. Scans backward looking for an entry containing a level (CATE not equal to 0).
    - 1. Terminates if scan is successful.
  - 2. Terminates the scan if it reaches an entry number which is less than that number specified in item FIRST (scanned to the beginning of the statement).

## 12. a. Name - LENGTH

b. Description - This procedure scans the CAT table looking for a comma or a dollar sign.

## c. Parameters

## 1. Input

1. PI1 - Contains the entry number in the CAT table of the starting position for the scan.

## 2. Output

1. LNCH - Set to the number of entries scanned by the procedure.

2. PI2 - Specifies the terminal entry in the CAT table of the expression scanned.

## d. Local items

## 1. T1

## e. Express items

## 1. CAT table items.

## g. Used by:

## 1. MB

## 2. DESIG

## h. Operations

1. Scan forward in the CAT table.

2. Use the push-down counter T1 to recognize when the procedure is scanning within a bracketed expression.

1. The scanning process ignores all entries within a bracketed expression.

3. Searches for a comma, \$, or a ) preceded by a \$.

1. Successful search.

12. h. 3. 1. 1. The number of entries examined is computed.

2. The procedure exits.

2. Unsuccessful search.

1. An error message is printed.

2. Procedure exits.

(If 250 entries are examined without success, the search is considered unsuccessful.)

## 13. a. Name - LVLCP

b. Description - This procedure attempts to locate an 'anchor-point'.

## c. Parameter

## 1. Output

1. LEV - 0 means two levels were found but no 'anchor-point' was determined; 1 means two levels were found which did determine an 'anchor-point'; 2 means the end-of-statement was encountered thus making an 'anchor-point'; 3 means the scan proceeded to the beginning of the statement without encountering a level.

## e. Express items

1. LEFT
2. RIGHT
3. RSLT
4. CAT table items

## f. Procedures called

1. LLOC

## g. Used by:

1. ANCHR

## h. Operations

1. Commence scan at the entry specified in the item RIGHT.
  2. Use the procedure LLOC to: locate an operator with a level or to recognize that there are no more operators within the statement.
  3. If the procedure recognizes that it is commencing outside of the statement (RIGHT is greater than LAST, i.e., at the \$).
1. If LLOC stopped on an 'OF' (CATC EQ 13, CATF EQ 5).
    1. If RSLT does not equal zero (meaning that a temporary working storage assignment could be made).
      1. The temporary assignment is forced by setting RIGHT equal to the value of RIGHT minus 1, setting LEV = 0.
      2. Exit.



13. h. 3. 2. If RSLT equals 0
  1. LEV is set to 2.
  2. Exit.
4. If the procedure recognizes that no other levels were encountered by the procedure LLOC (LEFT is less than FIRST).
  1. LEV is set to 3.
  2. If the procedure started with an entry containing a comma or a right bracket having a level greater than 20.
    1. This level is cleared to 0 (this type of bracket is only to be used as an 'anchor-point').
    2. If a temporary working storage assignment can be made (RSLT does not equal 0).
      1. LEV is changed to 0.
  3. Exit.
5. If the level on the operator located by procedure LLOC is less than the level on the operator specified in RIGHT (thus indicating no 'anchor-point')
  1. LEV is set to 0.
  2. Processing proceeds with h.4.2.
6. If the level of the operator located by procedure LLOC is greater than the level specified by RIGHT (this determines an 'anchor-point')
  1. A temporary storage assignment is forced if such an assignment can be made (RSLT did not equal 0). This forcing is done under two conditions: Condition 1: The operator on the left, i.e., the operator located by the procedure LLOC, describes an OF, or Condition 2: The operator on the left and the operator on the right, i.e., the operator with which the procedure started, are both relational operators.
    1. LEV is set to 0.
    2. RIGHT is set to a value of RIGHT minus 1.

10 July 1962

251

TM-555/020/00

13. h. 6. 1. 3. Exit.

2. LEV is set to 1.

3. Procedure exit.

14. a. Name - NTOI
- b. Description - This procedure obtains the name of a variable described in a DICT table entry from the item IDNS and places it into the item IMAGE.
- c. Parameters
  1. PI1 - Contains the DICT entry number of the variable to be processed.
  2. PI2 - Contains the starting character position in IMAGE into which the name is to be placed..
- e. Express items
  1. IMAGE
  2. IDNS
  3. DICT table items
- f. Procedures called
  1. BTOD
- g. Used by:
  1. PRTIL
  2. ERROR
  3. WORST
- h. Operations
  1. If the DICT entry does not specify a name (FCHB EQ 0 or NCHB EQ 0)
    1. Substitute the DICT entry number for the name.
    2. Exit.
  2. Obtain the name (or the first 10 characters thereof) and put it into IMAGE.

- 15. a. Name - OPND
- b. Description - This procedure converts an entry of the CAT table specifying a dictionary referenced variable or a subscript into an IL operand entry.
- c. Parameters
  - 1. Input
    - 1. P1 - Entry number of the IL table
    - 2. P2 - Entry number of the CAT table
- e. Express items
  - 1. IL table items
  - 2. CAT table items
- g. Used by:
  - 1. MB
  - 2. DESIG
  - 3. FACT
- h. Operations
  - 1. Determine the proper setting of ILC from the class (CATC) of the CAT table entry.
  - 2. Set the rest of the items in the IL entry from their corresponding items in the CAT entry.

## 16. a. Name - OPTM (Boolean Function)

b. Description - This procedure is used to determine the possibility of eliminating a GOTO, TEST or RETURN statement or optimizing a transfer implied by such a statement by effecting the implied transfer through the alteration of transfer points contained in IL 'GOTO' entries having generated labels as transfer points or IL 'relational' entries.

## c. Parameters

## 1. Input

1. PI1 - Specifies the label (either by number for generated labels or by DICT entry number for program labels) to be considered in the optimizing process.

2. PI2 - Identifies the type of label contained in PI1.

0 = statement label

1 = generated label

## 2. Output

1. OPTM - Indicates when a GOTO, TEST or RETURN statement can be eliminated.

0 = do not eliminate

1 = eliminate

## d. Local items

1. NLBL

2. GO

3. GOI

4. IFOI

## e. Express items

1. NIL

2. IL table items

## g. Used by:

1. MP

## h. Operations

16. h. 1. Perform a backward scan through the IL table examining operator entries (ILC = 0).
  1. The scan commences with the entry specified by the item NIL plus 1 (thus permitting 'information' entries inserted in the IL just prior to the operation of this procedure to be considered).
  2. For 'relational' entries (ILO LS 7)
    1. If the entry has not been involved in an optimizing process previously (ILR EQ 0) and if the only other entries (if there were any) thus far encountered in the scan were 'Information' entries
      1. The operator is complemented

LS set to GQ	GR set to IQ
EQ set to NQ	NQ set to EQ
IQ set to GR	GQ set to LS
      2. The transfer point in PI1 and PI2 is substituted for the original transfer point.
      3. ILR is set to 1 (to prevent double optimizing).
      4. OPTM is set to 1.
    2. Exit.
  3. For 'information' entries (ILO EQ 34)
    1. If the preceding entry contains a label (ILO EQ 16)
      1. Examine the region of the IL specified in the 'Information' entries.
        1. Substitute the transfer point in PI1 and PI2 for each transfer point specifying the label associated with the 'Information' entries.
        2. Clear the entries containing the information and its associated label.
  4. For 'GOTO' entries (ILO EQ 15)
    1. If the transfer point included in the entry describes a statement label, program label or generated label.

10 July 1962

256

TM-555/020/00

16. h. 1. 4. 1. 1. Set item GO to 1 (as a possible indication of two GOTO statements in a row).
2. Set item GOI to 1 indicating a non-label entry was examined.
5. For 'Dummy TST' entries (ILO EQ 25, ILE EQ 0).
  1. Proceed as in h.1.4.2.
6. For 'Label' entries (ILO EQ 16).
  1. Step counter NLBL.
7. For all other operator entries
  1. If no labels have been encountered (NLBL EQ 0) and a GOTO entry was encountered (GO EQ 1)
    1. Set OPTM to 1.
  2. Exit.

17. a. Name - POSOD
- b. Description - This procedure makes a backward scan thru the CAT table searching for the beginning of an operand to be placed in the IL.
- c. Parameters
  1. Output
    1. ODP - Specifies the CAT table entry at which the scan was terminated.
- e. Express items
  1. MID
  2. FIRST
  3. CAT table items
- f. Procedures called
  1. ERROR
- g. Used by:
  1. ANCHR
- h. Operations
  1. The procedure commences its scan at the entry number specified in the item MID.
  2. If the scan proceeds to an entry less than that entry specified in the item FIRST (scanned to the beginning of the statement)
    1. Output error #30.
    2. Exit.
  3. If a bracketer is encountered as the first entry examined
    1. If the bracketer is a right bracket
      1. The scan looks for an entry describing a subscripted variable (CATC EQ 5).
      2. Proceed at h.4.1.



10 July 1962

258

TM-555/020/00

17. h. 3. 2. If the bracketer encountered is a right parenthesis
  1. The scan proceeds to search for a bead-type entry (CATC EQ 16)
  2. Proceeds at h.4.1.
4. For all other cases
  1. ODP is set to the value of the last channel examined minus 1.
  2. The procedure exits.

10 July 1962

259

TM-555/020/00

- 18. a. Name - PPTCT
- b. Description - This procedure formats and outputs entries from the CAT table.
- e. Express items
  - 1. IMAGE
  - 2. CAT table items
- f. Procedures called
  - 1. BTOD
  - 2. PRNT
- g. Used by:
  - 1. ANCHR
- h. Operations
  - 1. Clears IMAGE.
  - 2. Outputs title.
  - 3. Formats entries into item IMAGE.
  - 4. Outputs IMAGE.

19. a. Name - PRNT
- b. Description - This procedure outputs the item IMAGE into the file DEBUG.
- c. Express items
1. IMAGE
- f. Procedures called
1. HOLLS
- g. Used by:
- |          |          |
|----------|----------|
| 1. PRTIL | 4. PPTCT |
| 2. PRTSR | 5. WORST |
| 3. PRTQ  |          |
- h. Operations
1. Using procedure HOLLS, convert the item IMAGE from STC to Hollerith code.
  2. Output the item IMAGE.
  3. Wait until the outputting is finished.
  4. Clear item IMAGE to blank characters.

20. a. Name - PRTHD
- b. Description - This procedure outputs a formatted heading for the IL, SRT and DRQ tables and the starting level of working storage.
- c. Parameters
1. Input
    1. FIL - Specifies the heading to be printed.
      - 0 = IL heading
      - 1 = SRT heading
      - 2 = DRQ heading
      - 3 = working storage heading
- e. Express items
1. IMAGE
- f. Procedures called
1. PRNT
- g. Used by:
1. WORST
  2. MB
- h. Operations
1. Clear IMAGE.
  2. Force heading to appear at the top of a new page.
  3. Put heading into IMAGE.

## 21. a. Name - PRTIL

b. Description - This procedure formats into item **IMAGE** entries from the IL table for printing.

## c. Parameters

## 1. Input

1. PI1 - Specifies the IL entry number of the first entry to be processed.
2. PI2 - Specifies the IL entry number of the last entry to be processed.

## e. Express items

- |         |                   |          |
|---------|-------------------|----------|
| 1. STIO | 4. STIB           | 7. IMAGE |
| 2. STIC | 5. PRCD           |          |
| 3. STIH | 6. IL table items |          |

## f. Procedures called

- |         |          |
|---------|----------|
| 1. ILDF | 4. PRINT |
| 2. BTOD |          |
| 3. NTOI |          |

## g. Used by:

1. MP

## h. Operations

1. Clear **IMAGE**
2. For operator type entries (ILC = 0)
  1. Format entry into left half of **IMAGE** (characters 1 - 35).
3. For operand, bead and subscript entries (ILC = 1,2,3).
  1. Format entry into right half of **IMAGE** (characters 36 - 71).
4. Put the entry number into **IMAGE**.
5. If the IL represents a procedure (PRCD NQ 9000).

10 July 1962

263

TM-555/020/00

21. h. 5. 1. Put the name of the procedure into IMAGE.

6. Output IMAGE using PRINT.

- 22. a. Name - PRTQ
- b. Description - This procedure formats into item IMAGE entries from the DRQ table.
- c. Parameters
  - 1. PI1 - Contains the DRQ entry number of the first entry to be processed.
  - 2. PI2 - Contains the DRQ entry number of the last entry to be processed.
- e. Express items
  - 1. STQA
  - 2. IMAGE
  - 3. DRQ table items
- f. Procedures called
  - 1. BTOD
  - 2. PRNT
- g. Used by:
  - 1. MB
- h. Operations
  - 1. Clear item IMAGE.
  - 2. Format an entry into IMAGE.
  - 3. Output IMAGE using PRNT.

10 July 1962

265

TM-555/020/00

23. a. Name - PRTER

b. Description - This procedure formats into item IMAGE entries from the SRT table.

c. Parameters

1. Input

1. PI1 - Contains the SRT entry number of the first entry to be processed.

2. PI2 - Contains the SRT entry number of the last entry to be processed.

e. Express items

1. IMAGE

2. STSD

3. SRT table items

f. Procedures called

1. BTOD

2. PRNT

g. Used by:

1. MB

h. Operations

1. Clear item IMAGE.

2. Format an entry into IMAGE.

3. Output IMAGE using PRNT.



## 24. a. Name - PUTIN

b. Description - This procedure generates IL operand entries from simple operands described in the CAT table. It also recognizes complex operands, either subscripted or beaded variables, and used procedure BDSV to enter these complex operands into the IL.

## c. Parameters

## 1. Input

1. CATPS - Specifies the starting CAT table entry of the expression to be processed.

2. DELET - Indicates whether the entries examined are to be cleared.

1 = do not clear  
0 = clear

## 2. Output

1. XXX - Error exit.

## d. Local items

1. CTPOS  
2. BDIND

## e. Express items

1. NIL	4. BNWS
2. NWS	5. CAT table items
3. RSLTI	6. IL table items

## f. Procedure called

1. ERROR  
2. BDSV

## g. Used by:

1. ANCHR

## h. Operations

1. Examine each entry by class (CATC) using the switch CLASS.

24. h. 1. 1. For entries containing statement labels (CATC EQ 1)
1. If the next CAT table entry describes a left bracket (CATC EQ 10, CATF EQ 3) meaning this label is a switch.
    1. Use procedure BDSV to process the expression.
    2. Exit.
  2. Generate an IL operand entry from the information in the CAT table entry.
  3. Exit.
2. For entries describing a file, constant, simple variable, or procedure (CATC EQ 2,3,4,6)
1. Generate an IL operand entry from the information in the CAT table (ILD = 0).
  2. Exit.
3. For entries describing a subscripted variable (CATC EQ 5)
1. Use procedure BDSV to process the expression.
  2. Exit.
4. For entries describing a subscript (CATC EQ 7)
1. Generate an IL operand entry describing the subscript (ILD = 2).
  2. Exit.
5. For entries containing brackets (CATC EQ 10)
1. Clear the entry if it describes a parenthesis.
6. For entries describing a bead (CATC EQ 16)
1. Use procedure BDSV to process the expression.
  2. Exit.
7. For entries describing a temporary storage (CATC EQ 18)

24. h. 1. 7. 1. Alter the assignment to specify result if possible (CATF EQ NWS and RSLTI LS 3).
  1. Set BNWS = 1 to acknowledge the change.
  2. Generate an IL operand entry specifying the temporary storage (ILD = 3).
  3. Exit.
8. For entries describing a generated label (CATC EQ 19)
  1. Generate an IL operand entry describing the generated label (ILD = 1).
  2. Exit.
9. For entries describing a direct code assignment operand (CATC EQ 20)
  1. Generate an IL operand entry specifying a temporary storage of result (ILD = 3, ILF = 0).
  2. Set ILE = CATF specifying the number of bits to the right of the binary point (all ones for floating point).
  3. Set ILY = 1 indicating that this entry was generated from a direct code assignment.
  4. Exit.
10. For entries describing other classes
  1. Output error #32.
  2. Exit to the error exit.

## 25. a. Name - SRCHL

b. Description - This procedure makes a backward scan through a statement searching for a logical operator or an IF, IFEITH, or an ORIF sequential operator.

## c. Parameters

## 1. Input

1. LOCC - Contains the starting CAT table entry number for the scan.

## 2. Output

1. LRSCH - Indicates if the scan was successful.

1 = successful  
0 = unsuccessful

2. XXX = error exit.

## d. Local items

1. LOC

## e. Express items

1. FIRST
2. CAT table items

## g. Used by:

1. ANCHR

## h. Operations

1. The boundaries for the scan are specified in items LOCC (the starting entry number) and FIRST (the terminating entry number).
2. Examine each entry by class (CATC).

1. If an entry contains a sequential operator or a logical operator.

1. LSRCH is set to 1.

- 25. h. 2. 1. 2. Exit
  - 2. If an entry contains an equal or exchange operator
    - 1. The scan is terminated unsuccessfully.
  - 3. If an entry contains a left parenthesis
    - 1. If the entry is preceded by a right bracket or an "OF"  
(CATC EQ 13, CATF EQ 5)
      - 1. The scan terminates unsuccessfully.
      - 2. The scan continues.
  - 4. If other than the above mentioned operators are encountered during the scan, the error exit is taken.

26. a. Name - SRCHR
- b. Description - This procedure makes a forward scan thru a JOVIAL statement or phrase searching for a logical operator.
- c. Parameters
1. Input
    1. LOCC - Contains the starting CAT table entry number for the scan.
  2. Output
    1. RSRCH - Indicates if the scan was successful. 0 means unsuccessful, 1 means successful, 2 means unsuccessful and that the scan was terminated at a \$, or else that the scan was commenced from within a procedure call.
    2. XXX - error exit.
- d. Local items
1. LOC
  2. PRI
  3. BKI
- e. Express items
1. CAT table items
  2. LAST
- g. Used by:
1. ANCHR
- h. Operations
1. If the procedure scans to an entry number which is greater than the entry specified in LAST (scanned to the end of the statement)
    1. RSRCH is set to 2.
    2. Exit.
  2. Examine each entry by class (CATC).

## 26. h. 2. 1. Analysis of bracketers (CATC EQ 10)

1. Use the push-down counter BKI and the parenthesis indicator PRI to:
  1. Recognize the right parenthesis terminating a procedure call or terminating the object of a bead (meaning that the procedure has commenced its scan within a procedure call or within the object of a bead).
    1. Set RSRCH equal to 2.
    2. Exit.
  2. Recognize when the procedure is scanning within a subscript expression or within a procedure call.
    1. All entries within a subscript expression or a procedure call are ignored.
2. If a dollar sign is encountered in the scan (CATC EQ 9, CATF EQ 4)
  1. RSRCH is set to 2.
  2. Exit.
3. If an OF is encountered (meaning the scan commenced within a procedure call) (CATC EQ 13, CATF EQ 5)
  1. PRI is set to 1 (so that the parenthesis around the call and the contents of the call will be ignored).
4. If a logical operator is encountered (CATC EQ 11)
  1. RSRCH is set to 1 (meaning that the scan is successful).
  2. Exit.
5. All other classes of entries encountered outside of bracketed expressions cause the error exit to be taken.

## 27. a. Name - SSPRO

b. Description - This procedure recognizes standard subscript expressions.  
(See procedure ANCHR for a definition of standard subscript expression.)

## c. Parameters

## 1. Input

1. NSS - Specifies the SS entry associated with the expression.

## 2. Output

1. STDI - Indicates whether the expression is of standard form.

1 = standard form

0 = non-standard form

2. COMI - Indicates whether the expression terminated with a  
'comma'.

1 = terminated with a 'comma'

0 = terminated with a non-comma

## d. Local items

## 1. TSS

## e. Express items

## 1. POS

## 2. ASPAC

## 3. SS table items

## 4. ACTS table items

## 5. CAT table items

## f. Procedures called

## 1. ERROR

## g. Used by:

## 1. ANCHR

## h. Operations

1. If the first entry of the expression describes a bead (CATC EQ 16)

1. Set STYPE = 1 for bead.



27. h. 1. 2. Exit.

2. Perform a right scan thru the expression examining each entry using switch CLSW.

1. For entries describing constants (CATC EQ 3)

1. If a previous constant has been encountered (SCKI EQ 1)

1. Exit.

2. Set SCKI = 1.

2. For entries describing simple variables (CATC EQ 4)

1. If a previous simple variable has been encountered (SSVI EQ 1) or a constant (SCKI EQ 1)

1. Exit.

2. Set SSVI = 1.

3. For entries describing subscripted variables (CATC EQ 5)

1. If the subscripted variable thus far fits one of the canonical forms

1. Set STYPE = 2.

2. Continue as a variable (h.2.2.1.)

4. For entries describing subscripts (CATC EQ 7)

1. If the subscript is inactive (ACTSS NQ 1)

1. Output error #34.

2. Continue as a variable (h.2.2.1.)

5. For entries describing separators (CATC EQ 9)

1. If the separator is not a 'comma'

1. Exit.

2. Set COMI = 1.

27. h. 2. 5. 3. Continue processing as a right bracket h.2.6.3.1.
6. Brackets (CATC EQ 10) are analyzed using switch BRKSW.
1. For entries describing left parenthesis (CATF EQ 0)
    1. If a previous left parenthesis has been encountered (SLPI EQ 1)
      1. Exit.
    2. Set SLPI = 1.
  2. For entries describing right parenthesis (CATF EQ 1)
    1. If a previous right parenthesis has been encountered (SRPI EQ 1) or an ABS modifier has not been encountered (SAVI NQ 1) or the ABS cannot be located with information in the SS entry
      1. Exit.
    2. Set SRPI = 1.
  3. For entries describing a right bracket (CATF EQ 3)
    1. If a right parenthesis has been encountered (SRPI EQ 1) and no ABS modifier (SAVI EQ 0)
      1. Exit.
    2. Set STDI = 1.
    3. Set POS to the entry last examined.
    4. Exit.
  7. For entries describing arithmetic operators (CATC EQ 13)
    1. If the operator is a '+', '-', or 'unary minus' (CATF EQ 0,1,6)
      1. If a previous arithmetic operator has been encountered (SPMI EQ 1)
        1. Exit.

10 July 1962

276

TM-555/020/00

27. h. 2. 7. 1. 2. Set SPMI = 1.

2. If the operator is 'ABS'

1. If a previous 'ABS' has been encountered (SAVI EQ 1)

1. Exit

1. Set SAVI = 1.

2. Set AVPOS = entry number describing the ABS.

8. For all other entries

1. Set STYPE = 0 (arithmetic expression).

2. Exit.

28. a. Name - STLE
- b. Description - This procedure generates a label IL operand entry (ILO = 16).
- c. Parameters
1. Input
    1. PI1 = Specifies the type of label to be put into the entry.  
  
1 = Generated label  
0 = Statement label
    2. PI2 - Contains: (1) the DICT entry number of the label if PI1 EQ 0, or (2) the actual number of the generated label if PI1 EQ 1.
- e. Express items
1. NIL
  2. OLAY
  3. SIND
  4. IL table entries
- g. Used by:
1. MP
  2. DESIG
  3. ANCHR
- h. Operations
1. If the next entry in the IL is an information operator entry (ILC EQ 0, ILO EQ 34)
    1. Increase the number of entries in the IL table by 2 (so that the Information entries will not be destroyed).
  2. Make up an IL label entry having ILO = 16, ILD = PI1, ILF = PI2.
  3. If the type of label is statement label and the label has not been used in a switch (OLAY EQ 0)
    1. Set SIND = 1 (If SIND is still set to 1 at the end of the program, there have been only forward transfers to this label.)

## 29. a. Name - WORST

b. Description - This procedure optimizes the amount of temporary storage required by the procedures of a program.

## c. Parameters

## 1. Input

1. AA - Specifies the type of processing to be done by WORST.

1 = perform the processing required by a procedure call.

2 = perform the processing required at the end of a procedure or the MP.

3 = perform the processing required at the end of the whole program.

4 = perform the initial housekeeping for the procedure.

## d. Local items

1. EXCP	4. FULL table items	7. MPWS
2. NOW	5. LWC tab items	8. MPNL
3. FWS table items	6. NLWC	9. WCNT

## e. Express items

1. PRCD	4. N'TREQ
2. NWS	5. IMAGE
3. MAXWS	6. TREQN

## f. Procedures called

1. PRNT  
2. BTOD  
3. NTOI

## g. Used by:

1. MP  
2. ANCHR

## h. Operations

Note: The procedure builds up entries in a table specifying the

maximum amount of working storage required each time a procedure call is made and in what procedure or MP the call was made from. The procedure then performs an analysis of this table and from the amount of working storage required at a procedure call and the nesting of linking of calls to a procedure from other procedures, determines the minimum amount of working storage required by a given procedure.

29. h. 1. If the type of processing is 'housekeeping' (AA EQ 4)
1. Clear local tables and items.
  2. Exit.
2. If the capacity of one of the local tables has been exceeded (EXCP EQ 1)
1. Exit.
3. Operate the switch JOB to determine the type of processing to be performed.
1. If the type of processing is that required by a procedure call (AA EQ 1)
    1. Adjust the amount of temporary storage by 2 (NOW = NWS-2) (temporary storage 1 and 2 is reserved for the Translators; therefore, all generator assigned temporary storage commences with 3).

Note: This procedure records in entries of the LWC table the amount of working storage required at a procedure call and from what procedure or MP the call was made. Since the procedure is only interested in the maximum amount of temporary storage for a procedure, it only records the maximum amount of temporary storage required by the several calls to a given procedure from a given procedure or MP, i.e., if ten calls to the same procedure are made from one procedure, only that call requiring the maximum amount of temporary storage is recorded in the LWC table.

2. If no previous entries have been made in the table (NLWC EQ 0)
  1. Continue at h.3.1.3.1.1.
3. Examine the other entries made during the processing of the current procedure or MP.

29. h. 3. 1. 3. 1. If there were no previous calls made during the current procedure or MP (PROS NQ PRCD)
  1. If making a new entry will exceed the capacity of the table
    1. Output an error message.
    2. Set EXCP = 1 to acknowledge the exceeding of a table.
    3. Exit.
  2. Make an entry in LWC including: the DICT entry number of the procedure being called (CAWL = CALL), the DICT entry number of the procedure or MP being processed (PROS = PRCD), and the amount of working storage (LOWS = NOW).
  3. Exit.
2. If a similar call has been made from the same procedure or MP (CAWL EQ CALL)
  1. If a larger amount of temporary storage is required by this latest call (NOW GR LOWS)
    1. Substitute the current amount for the preceding amount (LOWS = NOW).
  2. Exit.
2. If the type of processing is that required at the end of procedure or MP (AA EQ 2)
  1. If making a new entry in the PWS table will exceed its capacity
    1. Output an error message.
    2. Set EXCP = 1 to acknowledge the exceeding of the capacity.
    3. Exit.
  2. Make an entry into the PWS table including: the DICT entry number of the procedure or MP just processed

(PRON = PRCD), the maximum amount of working storage by the procedure or MP just processed (PRWS = MAXWS).

29. h. 3. 2. 3. Exit.

3. If the type of processing is that required at the end of the whole program (AA EQ 3)
  1. If there are no entries in LWC or PWS (NLWC EQ 0 or NPWS EQ 0)
    1. Set MAXWS = PRWS.
    2. Continue at h.3.3.7.
  2. Link the entries of the LWC table according to the following method:
    1. Locate the first entry recorded for a given procedure ( $PROS_n \text{ NQ } PROS_{n-1}$ ).
    2. Enter in LOCA the number of the first entry (located above 1.) in each entry specifying a call to that procedure (CAWL EQ PROS).
  3. Link the entries of the PWS table to the entries of the LWC table according to the following method:
    1. Enter in LOPR the PWS entry number of the entry describing the procedure which was called in LWC (CAWL EQ PRON).
  4. Through an analysis of the linking of the procedures, the minimum amount of temporary storage required by each procedure is determined. During the analysis, item WCNT is used to keep a running level of temporary storage at a given level of nesting of procedures. Table PNL is used in a push-down manner to regulate the processing of the different calls made from a procedure at a given level of nesting. A discussion of the processing is presented below using abbreviated LWS and PWS tables.



LWS Table						PWS Table	
Entry #	PROS	CAWL	LOCA	LOPR	LOWS	PRON	SLWS
1	MP	P1	4	2	4	MP	
2	MP	P2	6	3	2	P1	4
3	MP	P3	-	4	7	P2	5
4	P1	P2	6	3	1	P3	7
5	P1	P4	-	5	7	P4	11
6	P2	P4	-	5	5		

The calls are considered in the order in which they were made. Since the linkage of a given call is exhausted before the next call is considered, it is only necessary to examine the calls made from the MP.

29. h. 3. 3. 4. 1. MP calls P1.

1. Starting level of temporary storage (SLWS) for P1 = the level of temporary storage needed when the call was invoked (LOWS) = 4.
2. P1 calls P2.
  1. SLWS for P2 = the level when invoked (LOWS) + SLWS of P1 = 1+4 = 5.
  2. P2 calls P4.
    1. SLWS for P4 = level when invoked (LOWS) + SLWS of P2 = 5+5 = 10.
    2. No calls made in P4.
  3. P2 calls exhausted.
3. P1 calls P4.
  1. SLWS for P4 = level when invoked (LOWS) + SLWS of P1 = 7+4 = 11; previous SLWS for P4 was 10; substitute new SLWS for old (new SLWS OR old SLWS).
  2. No calls made in P4.

10 July 1962

283

TM-555/020/00

29. h. 3. 3. 4. 1. 4. P1 calls exhausted.
2. MP calls P2.
1. Calculated value not higher, so keep old SLWS for P2 = 5.
2. P2 calls P4.
1. SLWS for P4 =  $5+5 = 10$  as before.
2. No calls made in P4.
3. P2 calls exhausted.
3. MP calls P3.
1. SLWS for P3 = level when invoked (LWS) = 7.
2. No calls made in P3.
4. MP calls exhausted.
5. If the procedures are used recursively (one procedure calls a second which calls the first) indicated by the nesting level exceeding 20.
1. Output an error message.
2. Output the order of linkage.
3. Exit.
6. If there was at least one procedure in the program
1. If a list of the working storage levels is requested (PRST EQ 1)
1. Output the heading.
2. Set TREGN = SLWS of each procedure.
3. If the amount of temporary storage required by a procedure + its starting level exceeds the total required by the whole program (SLWS + PRWS GR MAXWS)
1. Set MAXWS = SLWS + PRWS.

29. h. 3. 3. 6. 4. Output the amount of temporary storage for each procedure if requested (PRST EQ 1).
7. If the amount of temporary storage for the MP exceeds that for any procedure (PRWS GR MAXWS)
1. Set MAXWS = PRWS.
8. Adjust the amount of temporary storage by 3 to account for 'result' (temporary storage of 0) and the 2 required by the Translators (MAXWS = MAXWS+3).
9. Set N'TREG = MAXWS.
10. Output the total amount of temporary storage if requested (PSRT EQ 1).
11. Exit.

## 30. a. Name - XFRPT

b. Description - This procedure determines the linkage (via transfer points) necessary in expressions involving relational operators, and adjusts IL operators and generates labels to effect the necessary linkage. (See SP-127, page 7.)

## c. Parameters

## 1. Input

1. LLOC - Specifies the CAT table entry last located by procedure LLOC (if LLOC is less than FIRST, the procedure is operating within a relational statement; if LLOC is greater than FIRST, the procedure is processing a Boolean assignment statement).
2. RRLOC - Specifies the CAT table entry of the last determined 'anchor-point'.
3. ILPOS - Specifies the IL table entry of the last relational operator entered into the IL.
4. RELAD - Indicates whether the 'anchor-point' is a relational or a logical operator.

1 = relational

0 = logical

## 2. Output

1. XXX - error exit

## d. Local items

1. RLOC
2. OPFRM
3. OPLVL

## e. Express items

- |         |         |                    |
|---------|---------|--------------------|
| 1. LAST | 4. FTP  | 7. IL table items  |
| 2. LBGR | 5. ATTP | 8. CAT table items |
| 3. TTP  | 6. AFTP |                    |

## f. Procedures called

1. ERROR

## 30. g. Used by:

## 1. ANCHR

## h. Operations

## 1. If the 'anchor-point' is an end-of-statement (LAST LS RRLOC)

## 1. If processing a normal relational (LLOC LS FIRST)

1. Generate a false transfer point label if there was not one generated earlier (FTP EQ 0).

2. Enter the false transfer point label into the transfer point of the entry specified in ILPOS.

3. Complement the operator specified in ILPOS.

4. Exit.

## 2. If processing a Boolean assignment (LLOC GR FIRST)

1. Continue processing at h.1.1.1., except use the alternate false transfer point (AFTP) instead of the false transfer point (FTP).

## 2. Set OPFRM to the type of operator described in the entry specified by RRLOC.

## 1. If the 'anchor-point' describes a relational operator (RELAD EQ 1).

1. OPFRM is set from CATA rather than from CATF.

## 3. Set OPLVL to the level (CATE) of the operator described in the entry specified by RRLOC.

## 1. If the 'anchor-point' describes a relational operator (RELAD EQ 1)

1. Set OPLVL to the level -1 (converts the relational level to a logical level).

## 4. Perform a right (forward) scan thru the expression commencing at the entry specified in RRLOC +1.

## 1. If the end-of-statement is encountered (RLOC GR LAST)

30. h. 4. 1. 1. If processing a normal relational statement (LLOC LS FIRST)
  1. If the type of logical operator is AND (OPFRM EQ 0)
    1. Proceed at h.1.
  2. If the type of logical operator is OR (OPFRM EQ 1)
    1. Generate a true transfer point label if one has not been previously generated (TTP EQ 0).
    2. Enter the true transfer point label into the transfer point of the IL entry specified by ILPOS.
    3. Exit.
2. If processing a Boolean assignment statement (LLOC GR FIRST)
  1. If the type of logical operator is AND (OPFRM EQ 0)
    1. Proceed at h.1.2.
  2. If the type of logical operator is OR (OPFRM EQ 1)
    1. Proceed at h.4.1.1.2. except use the alternate true transfer point (ATTP) instead of the true transfer point (TTP).
2. If the 'anchor-point' is a relational operator (RELAD EQ 1)
  1. If a relational operator is encountered in the scan which has a level less than that of the 'anchor-point' (CATE LS OPLVL)
    1. Set OPLVL to the level (CATE) of the relational operator just encountered.
3. If the scan encounters a logical operator which has a level less than that of the 'anchor-point' (CATE LS OPLVL)
  1. If the logical operator thus encountered is of the same type as the 'anchor-point' (CATF EQ OPFRM)
    1. Set OPLVL = CATE
    2. Continue the scan.

30. h. 4. 3. 2. If the logical operator thus encountered is not of the same type as the 'anchor-point' (CATF NQ OPFRM)
  1. If the class (CATC) of the succeeding entry is 0 (there was no label previously assigned to this operator)
    1. Generate a label and enter it in the next CAT table entry.
  2. Enter the label described in the next CAT table entry in the transfer point (ILF) of the IL entry specified by ILPOS.
  3. If the 'anchor-point' entry described an AND operator (CATF EQ 0)
    1. Complement the relational operator included in the IL entry specified by ILPOS.
4. Exit.

10 July 1962

289

TM-555/020/00

**E. ERROR MESSAGE DESCRIPTION**



- 101 - The statement commences with an illegal part of speech. In particular, the statement commenced with either a constant, logical operator, relational operator or arithmetic operator.  
Detected by: MP in region NEXT.
- 102 - The FOR statement to initialize a given subscript is encountered in the range of a FOR statement which previously initialized the same subscript.  
Detected by: MP in region L084.
- 103 - The 2 or 3 factor FOR statement is not terminated by a \$.  
Detected by: MP in region L084.
- 104 - An expression of the statement was examined which contained an odd number of parenthesis or brackets or which was not terminated by a comma or \$.  
Detected by: FACT or LENGTH.
- 105 - The TEST statement is used outside the range of any FOR statement.  
Detected by: MP in region L087.
- 106 - The TEST statement specifies a subscript activated by an 'incomplete' FOR statement.  
Detected by: MP in region L087.
- 107 - The TEST statement is used only in the range of an 'incomplete' FOR statement.  
Detected by: MP in region L087.
- 108 - An inactive subscript was specified by the TEST statement.  
Detected by: MP in region L087.
- 109 - An odd number of BEGINS or ENDS was discovered at the end of the program. Specifically, not enough ENDS were encountered to match the BEGINS.  
Detected by: MP in region L107.
- 110 - A non-label was encountered as a switch point in a switch declaration or as the object of a GOTO statement.  
~~Detected by: Procedure DESIG.~~
- 111 - A procedure was encountered as a switch point in a switch declaration or as the object of a GOTO statement.  
Detected by: Procedure DESIG.
- 112 - A declaration other than a switch, close or procedure was encountered by Gen2. (Gen1 is supposed to process all declarations other than those mentioned).  
Detected by: MP in region L14.

- 113 - The statement commenced with a separator other than a \$. (A dummy statement composed of just a \$ represents the only separator which Gen2 may encounter outside of statements).  
Detected by: MP in region L09.
- 114 - The relational operator 'EQ' is used in place of an '=' in an assignment statement, or else an IF, IFEITH, or ORIF operator is missing from a relational statement.  
Detected by: MP in region L02.
- 115 - The separator '=' was used in place of an 'EQ' in a relational statement or else a Boolean variable was used in an arithmetic expression.  
Detected by: MP in region L080.
- 116 - An inactive subscript was specified as a factor in the FOR statement.  
Detected by: Procedure FACT.
- 117 - An odd number of BEGINS or ENDS was discovered in the program. Specifically, too many ENDS were encountered to match the BEGINS.  
Detected by: MP in region L105.
- 118 - Some form of error was encountered in the analysis of a FOR factor. See a previous error message for the specific error discovered.  
Detected by: MP in region L084.
- 119 - An expression, rather than a whole statement, was encountered. Example: AA+BB/CC \$, rather than EE = AA+BB/CC \$.  
Detected by: MP in region L02.
- 120 - An odd number of BEGINS or ENDS was discovered in the program upon commencing this procedure. Specifically, too many BEGINS were encountered to match the ENDS. Check the BEGINS or ENDS in the preceding procedure or MP.  
Detected by: MP in region of L14.
- 121 - A bracketer has been used illegally in the statement. Specifically, either the statement commences with a parenthesis or bracket, a BEGIN, END, START, or TERM is located within the statement.  
Detected by: MP in region L10, Procedure SSFRO.
- 122 - Due to an excessive number of FOR statements in the program the capacity of an internal table of the generator has been exceeded. (SRT table).  
Detected by: MP in region L084.
- 123 - ~~(Not used)~~ All Next, Entry, or NWDSFN modifier missing
- 124 - (Not used)

- 125 - An illegal part of speech was encountered while processing this statement. Most likely this was caused by a missing \$ or previous error which caused part of a statement to be eliminated. Specifically, a BEGIN, END, START, or TERM bracketer was encountered within a statement, or a declaration, I/O operator or direct code operator was encountered within a statement. Detected by: ANCHR in the level assignment phase, SSPRO.
- 126 - Succeeding operators were encountered in the statement. Specifically, the \*,/,\*\* operators were encountered succeeding another operator. (Conversely, an operand is missing from the statement). Detected by: ANCHR in the level assignment phase.
- 127 - A Boolean variable was used illegally in the statement. Specifically, the variable was used in an arithmetic expression rather than a logical expression. Detected by: ANCHR in the level assignment phase while using procedures SRCHL and SRCHR.
- 128 - An odd number of parenthesis or brackets was encountered in the statement. Detected by: ANCHR in the level analysis phase.
- 129 - An absolute value modifier or a unary minus was used illegally in the statement. The generator was unable to determine the object of the absolute value modifier or the unary minus. Detected by: ANCHR in the level analysis phase.
- 130 - The absolute value of a bead expression was encountered in the statement. Example: ABS (BYTE [A,B] (ITEM'NAME')) Detected by: ANCHR in the level analysis phase.
- 131 - An exchange operator, '==', was encountered in a statement which was probably meant to be a Boolean assignment (=) statement. Detected by: ANCHR in the level analysis phase and XFRPT.
- 132 - An undetermined error was encountered in the statement. The error is probably due to a missing \$ which causes a part of speech to be used illegally. Detected by: ANCHR in the level analysis phase and by PUTIN and BDSV.
- 133 - A procedure was used as a function in the statement. Detected by: ANCHR in the analysis phase.
- 134 - A subscript was used outside of the range of its activating FOR statement. Detected by: ANCHR in the analysis phase and in SSPRO.
- 135 - One of the left operands for an operator in the statement could not be located. The operand was probably in the beaded variable of a bead expression. Detected by: POSOD.

- 136 - An excessive number of commas was encountered in a bead expression.  
Example: BYTE [A,B,C] (ITEM'NAME)  
Detected by: BGSV.
- 137 - An illegal part of speech was encountered within a subscript expression.  
Detected by: SSFRO.
- 138 - A non-Boolean item was encountered posing as a Boolean item in a relational statement.  
Detected by: MF in region L080.
- 139 - 200 errors detected by Gen2. Compilation terminated.
- 140 - The number of procedure calls exceeds the capacity permitted by the generator.
- 141 - The number of procedure declarations exceeds the capacity of the generator.
- 142 - Procedures in the program were used recursively.

Note: Statements are examined by procedure ANCHR for an operand-operator-operand-operator... pattern. Brackets and separators are determined from context to be either operands, operators or null. Errors 150, 151, 153, 154 are produced when a break is encountered in this pattern.

- 150 - While expecting an operand in the operand-operator pattern, an operator was encountered (succeeding operators). Specifically a sequential operator, declaration, input/output command or a BEGIN, END, START or TERM bracketer was encountered within the statement.  
Detected by: Procedure ANCHR.
- 151 - While expecting an operand in the operand-operator pattern, an operator was encountered (succeeding operators). Specifically, a separator, logical operator, relational operator, arithmetic operator (\* or /) or a subscript bracket or odd parenthesis was encountered within the statement.  
Detected by; Procedure ANCHR.
- 153 - While expecting an operator in the operand-operator pattern, an operand was encountered (succeeding operands). Specifically, a constant, odd parenthesis, unary minus or absolute value was encountered within the statement.  
Detected by: Procedure ANCHR.
- 154 - While expecting an operator in the operand-operator pattern, an operand was encountered (succeeding operands probably caused by a missing dollar sign). Specifically, a statement label, file, item, procedure, subscript, sequential operator, declaration, input/output command, bit/byte, direct code, direct assign or a BEGIN, END, START, TERM bracketer was encountered.  
Detected by: Procedure ANCHR.

UNCLASSIFIED

System Development Corporation,  
Santa Monica, California  
JOVIAL GENERATOR DESCRIPTION.  
Scientific rept., TM-555/020/00, by  
V. L. Cohen, M. H. Perstein.  
10 July 1962, 293p.

Unclassified report

DESCRIPTORS: Digital Computers. Machine  
Translation.

Identifiers: JOVIAL.

Describes Phase 1 and Phase 2 of the JOVIAL  
Generator. States that a compiler is a UNCLASSIFIED

---

UNCLASSIFIED

program which translates a higher  
level language called the source  
language, in which the solution to  
a problem is expressed, into a machine  
code representation of this solution  
called the object or target language.

UNCLASSIFIED

UNCLASSIFIED

System Development Corporation,  
Santa Monica, California  
MODIFICATION TO: TM-555/020/00,  
"JOVIAL GENERATOR DESCRIPTION"  
DATED 10 JULY 1962.  
Scientific rept., TM-555/020/00A,  
by M. H. Perstein.  
8 August 1962, 13p.

Unclassified report

DESCRIPTORS: Digital Computers.  
Machine Translation.

Identifiers: JOVIAL.

UNCLASSIFIED

---

UNCLASSIFIED

System Development Corporation,  
Santa Monica, California

MODIFICATION TO: TM-555/020/00,  
"JOVIAL GENERATOR DESCRIPTION"

DATED 10 JULY 1962.

Scientific rept., TM-555/020/00B,  
by V. L. Cohen, M. H. Perstein.  
18 September 1962, 21p.

Unclassified report

DESCRIPTORS: Digital Computers.  
Machine Translation.

Identifiers: JOVIAL.

UNCLASSIFIED

---



UNCLASSIFIED

System Development Corporation,  
Santa Monica, California

MODIFICATION TO: TM-555/020/00,  
"JOVIAL GENERATOR DESCRIPTION"

DATED 10 JULY 1962.

Scientific rept., TM-555/020/00C,  
by V. L. Cohen, M. H. Perstein.

16 October 1962, 5p.

Unclassified report

DESCRIPTORS: Digital Computers.  
Machine Translation.

Identifiers: JOVIAL

UNCLASSIFIED

---